Deterministic forecasting of short-crested ocean surface waves using the nonlinear Schrödinger equations

by

Svein Andreas Stokke Baardsen

THESIS

for the degree of

Master of Science in Computational Science and Engineering
Field of study Fluid Mechanics

(Master i Anvendt matematikk og mekanikk)
(Studieretning Fluidmekanikk)

Faculty of Mathematics and Natural Sciences
University of Oslo

May 2008

Det matematisk- naturvitenskapelige fakultet
Universitetet i Oslo

Preface

There is currently an interest in developing deterministic ocean wave forecasting procedures. This type of system is especially interesting for the offshore industry, where alot of operations are performed that are sensitive to surface displacement.

To forecast how an offshore structure will be influenced by surface waves there are three general steps. First one needs to measure the ocean surface some distance away from the structure, then calculate the evolution of the surface as it propagates towards the structure, and finally calculate the effect the waves will have on the structure. This thesis will concern itself with the second step of this procedure.

I will begin by developing the third order Schrödinger equations and solve them with regards to the surface amplitude in chapter 1. Explain methods that can be used to initialize the equations based on surface data sets in chapter 2. And finally test the procedure using different initial data sets in chapters 3-5.

I would like to thank: *Statoil* for making available the Draupner time series used in chapter 3; *OceanWaves*, *Seaflex* and *Statoil* for obtaining and allowing me to use the data sets in chapter 5; and especially my supervisor *Karsten Trulsen* for all his help and guidance.

Contents

List of Figures

Symbols and notation

A_{mn}	The m^{th} harmonic, n^{th} order amplitude of the velocity potential
B_{mn}	The m^{th} harmonic, n^{th} order amplitude of the reconstructed surface
c.c.	The complex conjugate of the preceding expression
$\mathbf{c}$	Phase velocity
$\mathbf{C}_g$	Group velocity
∇	Del operator: $\left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z}\right)$
∇_1	'Slow' del operator: $\left(\frac{\partial}{\partial x_1}, \frac{\partial}{\partial y_1}, \frac{\partial}{\partial z_1}\right)$
δ	Dirac function (aka impulse symbol)
ε	Wave steepness
η	Surface dispacement
$\tilde{\eta}$	Reconstructed surface dispacement
$\overline{\eta}$	The zeroth harmonic amplitude of the reconstructed surface (aka set-down)
g	Acceleration of gravity
$\mathbf{k}$	Wave vector: (k_x, k_y)
k	Wave number
ω	Angular velocity
ϕ	Velocity potential
$\overline{\phi}$	The zeroth harmonic amplitude of the velocity potential (aka slow drift)
$\mathbf{r}$	Horizontal position vector (x, y)
ρ	Mass density
t	Time coordinate
$\|\|X\|\|_p$	L_p norm of X
$\hat{X}$	Fourier transform of X
X^*	Complex conjugate of X
z	Vertical position coordinate

Chapter 1

The nonlinear Schrödinger equations

In this chapter I will derive the mathematical model that will be used to describe and predict the wave propagation.

1.1 The governing equations

We start with the continuity equation

$$\frac{\partial \rho}{\partial t} = -\nabla \cdot (\rho \mathbf{v})$$

and by assuming that the fluid is inviscid and incompressible and the flow is irrotational we can define the velocity as the gradient of the velocity potential $\mathbf{v} = \nabla\phi$ and we get the Laplace equation.

$$\nabla^2 \phi = 0$$

By assuming infinite depth we get the bottom condition

$$\lim_{z \to -\infty} \frac{\partial \phi}{\partial z} = 0$$

and to complete our system we have the surface conditions.

$$\text{Dynamic:} \qquad \frac{\partial \phi}{\partial t} + g\eta + \frac{1}{2}(\nabla\phi)^2 = 0 \qquad @ \quad z = \eta \qquad (1.\text{I})$$

$$\text{Kinematic:} \qquad \frac{\partial \eta}{\partial t} + \nabla\phi \cdot \nabla\eta = \frac{\partial \phi}{\partial z} \qquad @ \quad z = \eta \qquad (1.\text{II})$$

Unfortunelately this system does not have an analytic solution beside the trivial. So we will need to use some methods to transform this system into one that is solvable, the system that we will obtain is a version of what is known as the 3^{rd} order nonlinear Schrödinger equations.

1.1.1 Characteristic amplitude and frequency

There are a some defining characteristics that a wave field has, the most useful of which are related to how fast it moves and how long and high the waves are. These values are in general not completely constant but by assuming that they are we can gain gain ways to describe the wave fields.

For example; imagine we have a surface described by a simple sinusoidal wave with constant amplitude A and wavenumber k:

$$\eta = A\sin\left(kx - \omega t\right)$$

Now we would like to compute the average distance from equilibrium; this is accomplished by calculating the root-mean-square of the surface and since the wave is identical in each cycle it is sufficient to focus on a single cycle:

$$\begin{aligned}
\sqrt{\overline{\eta^2}} &= \sqrt{\frac{k}{2\pi}\int_0^{\frac{2\pi}{k}} \eta^2\, dx} \\
&= \sqrt{\frac{k}{2\pi}\int_0^{\frac{2\pi}{k}} A^2\sin^2\left(kx-\omega t\right)\, dx} \\
&= \sqrt{\frac{k}{2\pi}A^2\left(\frac{x}{2} - \frac{1}{2k}\sin\left(kx-\omega t\right)\cos\left(kx-\omega t\right)\right)\Big|_0^{\frac{2\pi}{k}}} \\
&= \frac{A}{\sqrt{2}}
\end{aligned}$$

So we see that to get our amplitude we need only multiply this with $\sqrt{2}$. A general wave field will not have a constant amplitude and so the value obtained by this method is referred to as the characteristic amplitude:

$$a_c = \sqrt{2\overline{\eta^2}}$$

The wave number k and angular frequency ω are connected to the wave length λ and period T through the equations $k = \frac{2\pi}{\lambda}$ and $\omega = \frac{2\pi}{T}$. The characteristic varieties ω_c and k_c of the variables are again the values that we obtain when we try to fit an actual surface to a simplified mathematical model.

We can then define the wave steepness as $\varepsilon = k_c a_c$, which will be a small number since λ is (almost always) significantly larger than a_c. We next require the surface elavation and velocity potential to be of similar magnitude and to explicitly show the magnitude of the terms we will replace all occurrences of ϕ and η with $\varepsilon\phi$ and $\varepsilon\eta$.

1.1.2 Taylor expansion

We begin by performing a taylor expansion about z=0 on the two surface conditions (1.I) and (1.II). Now since we are going to derive the 3$^{\text{rd}}$ order Schrödinger equations we do not need to worry about terms in the expansions whose sum of orders are larger than 3, so for instance ϕ will get the taylor expansion

$$\varepsilon\phi|_{z=\varepsilon\eta} = \varepsilon\phi|_{z=0} + \varepsilon^2\eta \left.\frac{\partial\phi}{\partial z}\right|_{z=0} + \varepsilon^3\frac{1}{2}\eta^2 \left.\frac{\partial^2\phi}{\partial z^2}\right|_{z=0} + \mathcal{O}\left(\varepsilon^4\right)$$

with similar expansions for all of ϕ's occurences. η is not affected by this as it is not a function of z. So with this change we now have the surface conditions:

$$\begin{aligned}\text{Dynamic:}\qquad & \varepsilon\frac{\partial\phi}{\partial t} + \varepsilon^2\eta\frac{\partial^2\phi}{\partial t\partial z} + \varepsilon^3\frac{1}{2}\eta^2\frac{\partial^3\phi}{\partial t\partial z^2} \\ & + \varepsilon^3\eta\nabla\phi\cdot\nabla\frac{\partial\phi}{\partial z} + \varepsilon g\eta + \varepsilon^2\frac{1}{2}\left(\nabla\phi\right)^2 + \mathcal{O}\left(\varepsilon^4\right) = 0 \qquad @ \quad z = 0\end{aligned}$$

$$\begin{aligned}\text{Kinematic:}\qquad & \varepsilon\frac{\partial\eta}{\partial t} + \varepsilon^2\nabla\phi\cdot\nabla\eta + \varepsilon^3\eta\nabla\eta\cdot\nabla\frac{\partial\phi}{\partial z} \\ & = \varepsilon\frac{\partial\phi}{\partial z} + \varepsilon^2\eta\frac{\partial^2\phi}{\partial z^2} + \varepsilon^3\frac{1}{2}\eta^2\frac{\partial^3\phi}{\partial z^3} + \mathcal{O}\left(\varepsilon^4\right) \qquad @ \quad z = 0\end{aligned}$$

1.1.3 Slow modulation variables

If we observe a wave train, we can see that the amplitudes of adjacent waves are indistinguishable and yet we can see that waves that are distant, in either time or space, from our wave, can have a vastly different amplitude. To model this behaviour of a function that seems to be constant on a short scale but is clearly not on a long scale we introduce the slow modulation

variables:

$$\begin{aligned}
t_n = \varepsilon^n t \qquad &\Rightarrow \qquad \frac{\partial f(t_0,t_1)}{\partial t} = \frac{\partial f(t_0,t_1)}{\partial t_0} + \varepsilon \frac{\partial f(t_0,t_1)}{\partial t_1} \\
x_n = \varepsilon^n t \qquad &\Rightarrow \qquad \frac{\partial f(x_0,x_1)}{\partial x} = \frac{\partial f(x_0,x_1)}{\partial x_0} + \varepsilon \frac{\partial f(x_0,x_1)}{\partial x_1} \\
y_n = \varepsilon^n t \qquad &\Rightarrow \qquad \frac{\partial f(y_0,y_1)}{\partial y} = \frac{\partial f(y_0,y_1)}{\partial y_0} + \varepsilon \frac{\partial f(y_0,y_1)}{\partial y_1} \\
z_n = \varepsilon^n t \qquad &\Rightarrow \qquad \frac{\partial f(z_0,z_1)}{\partial z} = \frac{\partial f(z_0,z_1)}{\partial z_0} + \varepsilon \frac{\partial f(z_0,z_1)}{\partial z_1}
\end{aligned}$$

We can now model a surface where the actual waves modulate with the fast '0' variables whilst the amplitude varies with the slow '1' variables. There are no conditions on the ε value that are needed for this to be correct so we will use the wave steepness ε. We can see that with these new variables, all our differential operations will be redefined. Since $t = t_0$ we can omit the 0 subscript and so we get a system with Laplace equation:

$$\begin{aligned}
\varepsilon \frac{\partial^2 \phi}{\partial x^2} + 2\varepsilon^2 \frac{\partial^2 \phi}{\partial x \partial x_1} + \varepsilon^3 \frac{\partial^2 \phi}{\partial x_1^2} + \varepsilon \frac{\partial^2 \phi}{\partial y^2} + 2\varepsilon^2 \frac{\partial^2 \phi}{\partial y \partial y_1} \\
+ \varepsilon^3 \frac{\partial^2 \phi}{\partial y_1^2} + \varepsilon \frac{\partial^2 \phi}{\partial z^2} + 2\varepsilon^2 \frac{\partial^2 \phi}{\partial z \partial z_1} + \varepsilon^3 \frac{\partial^2 \phi}{\partial z_1^2} + \mathcal{O}\left(\varepsilon^4\right) = 0
\end{aligned}$$

Bottom condition:

$$\lim_{z \to -\infty} \varepsilon \frac{\partial \phi}{\partial z} + \varepsilon^2 \frac{\partial \phi}{\partial z_1} = 0$$

And surface conditions:

$$\begin{aligned}
&\text{Dynamic:} \qquad \varepsilon \frac{\partial \phi}{\partial t} + \varepsilon^2 \frac{\partial \phi}{\partial t_1} + \varepsilon^2 \eta \frac{\partial^2 \phi}{\partial t \partial z} + \varepsilon^3 \eta \frac{\partial^2 \phi}{\partial t \partial z_1} + \varepsilon^3 \eta \frac{\partial^2 \phi}{\partial t_1 \partial z} + \varepsilon^3 \frac{1}{2} \eta^2 \frac{\partial^3 \phi}{\partial t \partial z^2} \\
&+ \varepsilon^3 \eta \nabla \phi \cdot \nabla \frac{\partial \phi}{\partial z} + \varepsilon g \eta + \varepsilon^2 \frac{1}{2} \left(\nabla \phi\right)^2 + \varepsilon^3 \nabla_1 \phi \cdot \nabla \phi + \mathcal{O}\left(\varepsilon^4\right) = 0 \qquad @ \quad z = 0
\end{aligned}$$

$$\begin{aligned}
&\text{Kinematic:} \qquad \varepsilon \frac{\partial \eta}{\partial t} + \varepsilon^2 \nabla \phi \cdot \nabla \eta + \varepsilon^3 \nabla_1 \phi \cdot \nabla \eta + \varepsilon^3 \nabla \phi \cdot \nabla_1 \eta + \varepsilon^3 \eta \nabla \eta \cdot \nabla \frac{\partial \phi}{\partial z} \\
&+ \varepsilon^2 \frac{\partial \eta}{\partial t_1} = \varepsilon \frac{\partial \phi}{\partial z} + \varepsilon^2 \frac{\partial \phi}{\partial z_1} + \varepsilon^2 \eta \frac{\partial^2 \phi}{\partial z^2} + \varepsilon^3 2\eta \frac{\partial^2 \phi}{\partial z \partial z_1} + \varepsilon^3 \frac{1}{2} \eta^2 \frac{\partial^3 \phi}{\partial z^3} + \mathcal{O}\left(\varepsilon^4\right) \ @ \quad z = 0
\end{aligned}$$

1.1.4 Form assumption

We now make the assumption that our waves can be represented by the harmonic functions below:

$$\phi = \varepsilon\bar{\phi} + \frac{1}{2}\left(Ae^{i(\mathbf{k}\cdot\mathbf{r}-\omega t)} + \varepsilon A_2 e^{2i(\mathbf{k}\cdot\mathbf{r}-\omega t)} + \varepsilon^2 A_3 e^{3i(\mathbf{k}\cdot\mathbf{r}-\omega t)} + \text{c.c.}\right) \quad \text{(1.III)}$$

$$\eta = \varepsilon\bar{\eta} + \frac{1}{2}\left(Be^{i(\mathbf{k}\cdot\mathbf{r}-\omega t)} + \varepsilon B_2 e^{2i(\mathbf{k}\cdot\mathbf{r}-\omega t)} + \varepsilon^2 B_3 e^{3i(\mathbf{k}\cdot\mathbf{r}-\omega t)} + \text{c.c.}\right) \quad \text{(1.IV)}$$

Here the amplitudes $\bar{\eta}$ and B_n are functions of the slow variables x_1, y_1 and t_1, the A_n are also functions of the fast z variable, but as we will see they decay exponentially on the vertical scale and so are not dependent on the slow z_1 variable. Finally $\bar{\phi}$ is is a function of all the slow variables but since the slow drift describes the flow that is not induced by the wave motion it is not dependent on the fast z. The function dependencies are as follows:

$$A_n = A_n(x_1, y_1, z, t_1) \qquad B_n = B_n(x_1, y_1, t_1)$$
$$\bar{\phi} = \bar{\phi}(x_1, y_1, z_1, t_1) \qquad \bar{\eta} = \bar{\eta}(x_1, y_1, t_1)$$

Our system now consists of the kinematic surface condition:

$$\begin{aligned}
&\varepsilon^2\frac{1}{4}\left(AB^*k^2 + A^*Bk^2 - B^*\frac{\partial^2 A}{\partial z^2} - B\frac{\partial^2 A^*}{\partial z^2}\right)\\
&\quad+\varepsilon^3\frac{1}{4}\left(iAk_x\frac{\partial B^*}{\partial x_1} + iAk_y\frac{\partial B^*}{\partial y_1} + iBk_x\frac{\partial A^*}{\partial x_1} + iBk_y\frac{\partial A^*}{\partial y_1} + 4\frac{\partial\bar{\eta}}{\partial t_1}\right.\\
&\qquad\left.-iA^*k_x\frac{\partial B}{\partial x_1} - iA^*k_y\frac{\partial B}{\partial y_1} - iB^*k_x\frac{\partial A}{\partial x_1} - iB^*k_y\frac{\partial A}{\partial y_1} - 4\frac{\partial\bar{\phi}}{\partial z_1}\right)\\
&\qquad+\left[\left(-\varepsilon\frac{1}{2}\left(\frac{\partial A}{\partial z} + i\omega B\right) + \varepsilon^2\frac{1}{2}\frac{\partial B}{\partial t_1}\right.\right.\\
&\quad\left.+\varepsilon^3\frac{1}{16}\left(2k^2B^2\frac{\partial A^*}{\partial z} - 4B_2\frac{\partial^2 A^*}{\partial z^2} + 8k^2A_2B^* + 8k^2A^*B_2\right.\right.\\
&\quad\left.\left.-B^2\frac{\partial^3 A^*}{\partial z^3} - 4B^*\frac{\partial^2 A_2}{\partial z^2} - 2|B|^2\frac{\partial^3 A}{\partial z^3} - 8B\frac{\partial^2\bar{\phi}}{\partial z^2} - 8\bar{\eta}\frac{\partial^2 A}{\partial z^2}\right)e^{i(\mathbf{k}\cdot\mathbf{r}-\omega t)}\right.\\
&\qquad\left.+\left(\varepsilon^2\frac{1}{4}\left(-k^2AB - B\frac{\partial^2 A}{\partial z^2} - 2\frac{\partial A_2}{\partial z} - 4i\omega B_2\right)\right.\right.\\
&\left.\left.+\varepsilon^3\frac{1}{4}\left(ik_xA\frac{\partial B}{\partial x_1} + ik_yA\frac{\partial B}{\partial y_1} + ik_yB\frac{\partial A}{\partial y_1} + ik_xB\frac{\partial A}{\partial x_1} + 2\frac{\partial B_2}{\partial t_1}\right)\right)e^{2i(\mathbf{k}\cdot\mathbf{r}-\omega t)}\right.\\
&\quad-\varepsilon^3\frac{1}{16}\left(24i\omega B_3 + B^2\frac{\partial^3 A}{\partial z^3} + 8k^2A_2B + 8k^2AB_2 + 2k^2B^2\frac{\partial A}{\partial z} + 8\frac{\partial A_3}{\partial z}\right.\\
&\quad\left.\left.+4B_2\frac{\partial^2 A}{\partial z^2} + 4B\frac{\partial^2 A_2}{\partial z^2}\right)e^{3i(\mathbf{k}\cdot\mathbf{r}-\omega t)} + \text{c.c.}\right] + \mathcal{O}\left(\varepsilon^4\right) = 0 \qquad @ \quad z = 0
\end{aligned}$$

Dynamic surface condition:

$$\begin{aligned}
&\varepsilon^2\frac{1}{4}\left(k^2\left|A\right|^2+4g\bar{\eta}+i\omega B\frac{\partial A^*}{\partial z}-i\omega B^*\frac{\partial A}{\partial z}+\frac{\partial A^*}{\partial z}\frac{\partial A}{\partial z}\right)\\
&\quad+\varepsilon^3\frac{1}{4}\left(B^*\frac{\partial^2 A}{\partial t_1\partial z}+B\frac{\partial^2 A^*}{\partial t_1\partial z}+ik_xA\frac{\partial A^*}{\partial x_1}+ik_yA\frac{\partial A^*}{\partial y_1}-ik_xA^*\frac{\partial A}{\partial x_1}\right.\\
&\qquad\left.-ik_yA^*\frac{\partial A}{\partial y_1}+4\frac{\partial\bar{\phi}}{\partial t_1}\right)+\left[\left(\varepsilon\frac{1}{2}\left(gB-i\omega A\right)+\varepsilon^2\frac{1}{2}\frac{\partial A}{\partial t_1}\right.\right.\\
&+\varepsilon^3\frac{1}{16}\left(8k^2A^*A_2-2k^2AB^*\frac{\partial A}{\partial z}+2k^2A^*B\frac{\partial A}{\partial z}+2k^2AB\frac{\partial A^*}{\partial z}+4\frac{\partial A^*}{\partial z}\frac{\partial A_2}{\partial z}\right.\\
&\quad+2B\frac{\partial A}{\partial z}\frac{\partial^2 A^*}{\partial z^2}-8i\omega B^*\frac{\partial A_2}{\partial z}+8\frac{\partial A}{\partial z}\frac{\partial\bar{\phi}}{\partial z}+i\omega B^2\frac{\partial^2 A^*}{\partial z^2}+2B\frac{\partial A^*}{\partial z}\frac{\partial^2 A}{\partial z^2}\\
&\quad\left.-8i\omega\bar{\eta}\frac{\partial A}{\partial z}-2i\omega\left|B\right|^2\frac{\partial^2 A}{\partial z^2}+2B^*\frac{\partial A}{\partial z}\frac{\partial^2 A}{\partial z^2}+4i\omega B_2\frac{\partial A^*}{\partial z}\right)\Bigg)e^{i(\mathbf{k}\cdot\mathbf{r}-\omega t)}\\
&+\left(\varepsilon^2\frac{1}{8}\left(+\frac{\partial A}{\partial z}^2-k^2A^2-2i\omega B\frac{\partial A}{\partial z}+4gB_2-8i\omega A_2\right)+\varepsilon^3\frac{1}{4}\left(ik_xA\frac{\partial A}{\partial x_1}\right.\right.\\
&\quad\left.+ik_yA\frac{\partial A}{\partial y_1}+B\frac{\partial^2 A}{\partial t_1\partial z}+\frac{\partial A_2}{\partial t_1}\right)\Bigg)e^{2i(\mathbf{k}\cdot\mathbf{r}-\omega t)}+\varepsilon^3\frac{1}{16}\left(4\frac{\partial A}{\partial z}\frac{\partial A_2}{\partial z}\right.\\
&\quad+2B\frac{\partial A}{\partial z}\frac{\partial^2 A}{\partial z^2}-2k^2AB\frac{\partial A}{\partial z}-i\omega B^2\frac{\partial^2 A}{\partial z^2}-4i\omega B_2\frac{\partial A}{\partial z}-8i\omega B\frac{\partial A_2}{\partial z}\\
&\left.\left.\left.-8k^2AA_2+8gB_3-24i\omega A_3\right)e^{3i(\mathbf{k}\cdot\mathbf{r}-\omega t)}+\text{c.c.}\right]+\mathcal{O}\left(\varepsilon^4\right)=0\qquad@\quad z=0
\end{aligned}$$

Laplace equation:

$$\begin{aligned}
&\varepsilon^2\frac{\partial^2\bar{\phi}}{\partial z^2}+2\varepsilon^3\frac{\partial^2\bar{\phi}}{\partial z\partial z_1}+\left[\left(\varepsilon\frac{1}{2}\left(\frac{\partial^2 A}{\partial z^2}-k^2A\right)\right.\right.\\
&\quad\left.+i\varepsilon^2\left(\frac{\partial A}{\partial x_1}k_x+\frac{\partial A}{\partial y_1}k_y\right)+\frac{1}{2}\varepsilon^3\left(\frac{\partial^2 A}{\partial y_1^2}+\frac{\partial^2 A}{\partial x_1^2}\right)\right)e^{i(\mathbf{k}\cdot\mathbf{r}-\omega t)}\\
&\quad+\left(\varepsilon^2\left(\frac{1}{2}\frac{\partial^2 A_2}{\partial z^2}-2k^2A_2\right)+2i\varepsilon^3\left(\frac{\partial A_2}{\partial y_1}k_y+\frac{\partial A_2}{\partial x_1}k_x\right)\right)e^{2i(\mathbf{k}\cdot\mathbf{r}-\omega t)}\\
&\qquad\left.\left.+\varepsilon^3\left(\frac{1}{2}\frac{\partial^2 A_3}{\partial z^2}-\frac{9}{2}k^2A_3\right)e^{3i(\mathbf{k}\cdot\mathbf{r}-\omega t)}+\text{c.c.}\right]+\mathcal{O}\left(\varepsilon^4\right)=0
\end{aligned}$$

And bottom condition:

$$\begin{aligned}
&\varepsilon^3\frac{\partial\bar{\phi}}{\partial z_1}+\left[\varepsilon\frac{1}{2}\frac{\partial A}{\partial z}e^{i(\mathbf{k}\cdot\mathbf{r}-\omega t)}+\varepsilon^2\frac{1}{2}\frac{\partial A_2}{\partial z}e^{2i(\mathbf{k}\cdot\mathbf{r}-\omega t)}\right.\\
&\qquad\left.+\varepsilon^3\frac{1}{2}\frac{\partial A_3}{\partial z}e^{3i(\mathbf{k}\cdot\mathbf{r}-\omega t)}+\text{c.c.}\right]+\mathcal{O}\left(\varepsilon^4\right)=0
\end{aligned}$$

1.1.5 Perturbation

The last addition to our system is to perturbate the functions. Since the radar images we have will allow us to determine B there is no need to perturbate that function. Moreover, since we are only interested in 3$^{\text{rd}}$ order terms and below there is no point in introducing variables like A_{31} and B_{22} since the additional ε terms would automatically push them over our cut-off point. The perturbation expansions are then:

$$\begin{aligned} A &= A_{10} + \varepsilon A_{11} + \varepsilon^2 A_{12} \\ A_2 &= A_{20} + \varepsilon A_{21} & B_2 &= B_{20} + \varepsilon B_{21} \\ \bar{\phi} &= \bar{\phi}_0 + \varepsilon \bar{\phi}_1 & \bar{\eta} &= \bar{\eta}_0 + \varepsilon \bar{\eta}_1 \end{aligned}$$

We now have our, quite expansive, Schrödinger equations represented by the dynamic surface condition:

$$\begin{aligned} &\varepsilon^2 \frac{1}{4}\left(k^2 |A_{10}|^2 - i\omega B^* \frac{\partial A_{10}}{\partial z} + \frac{\partial A_{10}^*}{\partial z}\frac{\partial A_{10}}{\partial z} + i\omega B \frac{\partial A_{10}^*}{\partial z} + 4g\bar{\eta}_0\right) \\ &\quad + \varepsilon^3 \frac{1}{4}\left(4\frac{\partial \bar{\phi}_0}{\partial t_1} + \frac{\partial A_{10}^*}{\partial z}\frac{\partial A_{11}}{\partial z} + k^2 A_{10} A_{11}^* + \frac{\partial A_{10}}{\partial z}\frac{\partial A_{11}^*}{\partial z} + B^* \frac{\partial^2 A_{10}}{\partial t_1 \partial z}\right. \\ &\qquad + B\frac{\partial^2 A_{10}^*}{\partial t_1 \partial z} - i\omega B^* \frac{\partial A_{11}}{\partial z} + i\omega B \frac{\partial A_{11}^*}{\partial z} + 4g\bar{\eta}_1 + k^2 A_{10}^* A_{11} \\ &\qquad \left. + ik_x A_{10}\frac{\partial A_{10}^*}{\partial x_1} + ik_y A_{10}\frac{\partial A_{10}^*}{\partial y_1} - ik_x A_{10}^*\frac{\partial A_{10}}{\partial x_1} - ik_y A_{10}^*\frac{\partial A_{10}}{\partial y_1}\right) \\ &+ \left[\left(\varepsilon\frac{1}{2}(gB - i\omega A_{10}) + \varepsilon^2\frac{1}{2}\left(\frac{\partial A_{10}}{\partial t_1} - i\omega A_{11}\right) + \varepsilon^3\frac{1}{16}\left(2B\frac{\partial A_{10}}{\partial z}\frac{\partial^2 A_{10}^*}{\partial z^2}\right.\right.\right. \\ &\qquad + 4i\omega B_{20}\frac{\partial A_{10}^*}{\partial z} - 8i\omega\bar{\eta}_0\frac{\partial A_{10}}{\partial z} + 2k^2 A_{10} B\frac{\partial A_{10}^*}{\partial z} + 2k^2 A_{10}^* B\frac{\partial A_{10}}{\partial z} \\ &\quad - 2k^2 A_{10} B^*\frac{\partial A_{10}}{\partial z} + 2B\frac{\partial A_{10}^*}{\partial z}\frac{\partial^2 A_{10}}{\partial z^2} + 2B^*\frac{\partial A_{10}}{\partial z}\frac{\partial^2 A_{10}}{\partial z^2} + 4\frac{\partial A_{10}^*}{\partial z}\frac{\partial A_{20}}{\partial z} \\ &\qquad + 8\frac{\partial A_{10}}{\partial z}\frac{\partial \bar{\phi}_0}{\partial z} - 8i\omega B^*\frac{\partial A_{20}}{\partial z} + 8\frac{\partial A_{11}}{\partial t_1} + i\omega B^2\frac{\partial^2 A_{10}^*}{\partial z^2} + 8k^2 A_{10}^* A_{20} \\ &\quad \left.\left.\left. -2i\omega |B|^2\frac{\partial^2 A_{10}}{\partial z^2} - 8i\omega A_{12}\right)\right) e^{i(\mathbf{k}\cdot\mathbf{r}-\omega t)} + \left(\varepsilon^2\frac{1}{8}\left(4gB_{20} - 2i\omega B\frac{\partial A_{10}}{\partial z}\right.\right.\right. \\ &\qquad \left. - k^2 A_{10}^2 + \left(\frac{\partial A_{10}}{\partial z}\right)^2 - 8i\omega A_{20}\right) + \varepsilon^3\frac{1}{4}\left(B\frac{\partial^2 A_{10}}{\partial t_1 \partial z} + \frac{\partial A_{10}}{\partial z}\frac{\partial A_{11}}{\partial z}\right. \\ &\qquad + ik_y A_{10}\frac{\partial A_{10}}{\partial y_1} + ik_x A_{10}\frac{\partial A_{10}}{\partial x_1} + 2\frac{\partial A_{20}}{\partial t_1} - 4i\omega A_{21} + 2gB_{21} \\ &\quad \left.\left.\left. -k^2 A_{10} A_{11} - iB\omega\frac{\partial A_{11}}{\partial z}\right)\right) e^{2i(\mathbf{k}\cdot\mathbf{r}-\omega t)} + \varepsilon^3\frac{1}{16}\left(2B\frac{\partial A_{10}}{\partial z}\frac{\partial^2 A_{10}}{\partial z^2} - 8k^2 A_{10} A_{20}\right. \\ &\qquad + 4\frac{\partial A_{10}}{\partial z}\frac{\partial A_{20}}{\partial z} - 4i\omega B_{20}\frac{\partial A_{10}}{\partial z} - 2k^2 A_{10} B\frac{\partial A_{10}}{\partial z} + 8gB_3 - 8i\omega B\frac{\partial A_{20}}{\partial z} \\ &\quad \left.\left. -24i\omega A_3 - i\omega B^2\frac{\partial^2 A_{10}}{\partial z^2}\right) e^{3i(\mathbf{k}\cdot\mathbf{r}-\omega t)} + \text{c.c.}\right] + \mathcal{O}\left(\varepsilon^4\right) = 0 \qquad @ \quad z = 0 \end{aligned}$$

Kinematic surface condition:

$$\varepsilon^2\frac{1}{4}\left(k^2A_{10}^*B+k^2A_{10}B^*-B\frac{\partial^2A_{10}^*}{\partial z^2}-B^*\frac{\partial^2A_{10}}{\partial z^2}\right)+\varepsilon^3\frac{1}{4}\left(-ik_yB^*\frac{\partial A_{10}}{\partial y_1}\right.$$
$$-4\frac{\partial\bar{\phi}_0}{\partial z_1}+4\frac{\partial\bar{\eta}_0}{\partial t_1}-ik_yA_{10}^*\frac{\partial B}{\partial y_1}+k^2A_{11}B^*-B\frac{\partial^2A_{11}^*}{\partial z^2}+ik_xB\frac{\partial A_{10}^*}{\partial x_1}$$
$$+ik_yB\frac{\partial A_{10}^*}{\partial y_1}-ik_xA_{10}^*\frac{\partial B}{\partial x_1}+ik_yA_{10}\frac{\partial B^*}{\partial y_1}-B^*\frac{\partial^2A_{11}}{\partial z^2}+ik_xA_{10}\frac{\partial B^*}{\partial x_1}$$
$$\left.+k^2A_{11}^*B-ik_xB^*\frac{\partial A_{10}}{\partial x_1}\right)+\left[\left(-\varepsilon\frac{1}{2}\left(\frac{\partial A_{10}}{\partial z}+i\omega B\right)\right.\right.$$
$$+\varepsilon^2\frac{1}{2}\left(\frac{\partial B}{\partial t_1}-\frac{\partial A_{11}}{\partial z}\right)+\varepsilon^3\frac{1}{16}\left(8k^2A_{10}^*B_{20}+8k^2A_{20}B^*-4B_{20}\frac{\partial^2A_{10}^*}{\partial z^2}\right.$$
$$+2k^2B^2\frac{\partial A_{10}^*}{\partial z}-8\bar{\eta}_0\frac{\partial^2A_{10}}{\partial z^2}-2\left|B\right|^2\frac{\partial^3A_{10}}{\partial z^3}-4B^*\frac{\partial^2A_{20}}{\partial z^2}-8\frac{\partial A_{12}}{\partial z}-B^2\frac{\partial^3A_{10}^*}{\partial z^3}$$
$$\left.\left.-8B\frac{\partial^2\bar{\phi}_0}{\partial z^2}\right)\right)e^{i(\mathbf{k}\cdot\mathbf{r}-\omega t)}+\left(\varepsilon^2\frac{1}{4}\left(-2\frac{\partial A_{20}}{\partial z}-k^2A_{10}B-B\frac{\partial^2A_{10}}{\partial z^2}-4i\omega B_{20}\right)\right.$$
$$+\varepsilon^3\frac{1}{4}\left(2\frac{\partial B_{20}}{\partial t_1}-2\frac{\partial A_{21}}{\partial z}+ik_xB\frac{\partial A_{10}}{\partial x_1}+ik_yB\frac{\partial A_{10}}{\partial y_1}+ik_xA_{10}\frac{\partial B}{\partial x_1}\right.$$
$$\left.\left.+ik_yA_{10}\frac{\partial B}{\partial y_1}-k^2A_{11}B-B\frac{\partial^2A_{11}}{\partial z^2}-4i\omega B_{21}\right)\right)e^{2i(\mathbf{k}\cdot\mathbf{r}-\omega t)}+\varepsilon^3\frac{1}{16}\left(-8\frac{\partial A_3}{\partial z}\right.$$
$$-B^2\frac{\partial^3A_{10}}{\partial z^3}-2k^2B^2\frac{\partial A_{10}}{\partial z}-4B\frac{\partial^2A_{20}}{\partial z^2}-4B_{20}\frac{\partial^2A_{10}}{\partial z^2}-24i\omega B_3$$
$$\left.\left.-8k^2A_{20}B-8k^2A_{10}B_{20}\right)e^{3i(\mathbf{k}\cdot\mathbf{r}-\omega t)}+\text{c.c.}\right]+\mathcal{O}\left(\varepsilon^4\right)=0\qquad@\quad z=0$$

Laplace equation:

$$+\varepsilon^3 2\frac{\partial^2\bar{\phi}_0}{\partial z\partial z_1}+\left[\left(\varepsilon\frac{1}{2}\left(\frac{\partial^2A_{10}}{\partial z^2}-k^2A_{10}\right)\right.\right.$$
$$+\varepsilon^2\frac{1}{2}\left(2ik_x\frac{\partial A_{10}}{\partial x_1}+\frac{\partial^2A_{11}}{\partial z^2}-k^2A_{11}+2ik_y\frac{\partial A_{10}}{\partial y_1}\right)+\varepsilon^3\frac{1}{2}\left(\frac{\partial^2A_{12}}{\partial z^2}-k^2A_{12}\right.$$
$$\left.\left.+\frac{\partial^2A_{10}}{\partial x_1^2}+\frac{\partial^2A_{10}}{\partial y_1^2}+2ik_x\frac{\partial A_{11}}{\partial x_1}+2ik_y\frac{\partial A_{11}}{\partial y_1}\right)\right)e^{i(\mathbf{k}\cdot\mathbf{r}-\omega t)}+\left(\varepsilon^2\frac{1}{2}\left(\frac{\partial^2A_{20}}{\partial z^2}\right.\right.$$
$$\left.-4k^2A_{20}\right)+\varepsilon^3\frac{1}{2}\left(\frac{\partial^2A_{21}}{\partial z^2}+4ik_x\frac{\partial A_{20}}{\partial x_1}+4ik_y\frac{\partial A_{20}}{\partial y_1}-4k^2A_{21}\right)\Big)e^{2i(\mathbf{k}\cdot\mathbf{r}-\omega t)}$$
$$\left.+\varepsilon^3\frac{1}{2}\left(\frac{\partial^2A_3}{\partial z^2}-9k^2A_3\right)e^{3i(\mathbf{k}\cdot\mathbf{r}-\omega t)}+\text{c.c.}\right]+\mathcal{O}\left(\varepsilon^4\right)=0$$

And bottom condition:

$$\varepsilon^3\frac{\partial\bar{\phi}_0}{\partial z_1}+\frac{1}{2}\left[\left(\varepsilon\frac{\partial A_{10}}{\partial z}+\varepsilon^2\frac{\partial A_{11}}{\partial z}+\varepsilon^3\frac{\partial A_{12}}{\partial z}\right)e^{i(\mathbf{k}\cdot\mathbf{r}-\omega t)}\right.$$
$$\left.+\left(\varepsilon^2\frac{\partial A_{20}}{\partial z}+\varepsilon^3\frac{\partial A_{21}}{\partial z}\right)e^{2i(\mathbf{k}\cdot\mathbf{r}-\omega t)}+\varepsilon^3\frac{\partial A_3}{\partial z}e^{3i(\mathbf{k}\cdot\mathbf{r}-\omega t)}+\text{c.c.}\right]+\mathcal{O}\left(\varepsilon^4\right)=0$$

1.2 The leading order method

Now that we have our equations we can begin the work of actually solving them. As mentioned we will use the radar images to determine B, therefore to solve the equations we will need to find a way to express the other functions in terms of B.

To solve the equations we gather all the terms according to the order of ε and harmonic modulation rate and require that terms of equal 'status' must balance eachother out.

1.2.1 $\mathcal{O}(\varepsilon)$

Since we have explicitly stated the dependence on the horisontal and temporal variables, $\mathbf{r}$ and t, in the form assumptions (1.III) and (1.IV) the Laplace equation is reduced to a homogenous ordinary differential equation that is easily solvable:

$$k^2 A_{10} = \frac{\partial^2 A_{10}}{\partial z^2} \qquad \Rightarrow \qquad A_{10} = A_{10}^{+} e^{kz} + A_{10}^{-} e^{-kz}$$

And by using the bottom condition we get

$$\lim_{z \to -\infty} \frac{\partial A_{10}}{\partial z} = 0 \qquad \Rightarrow \qquad A_{10} = A_{10}^{+} e^{kz}$$

Now we have two surface conditions which will give us two different expressions for A_{10}, by applying the dynamic condition we get:

$$\begin{aligned} i\omega A_{10} &= gB && @ \quad z = 0 \\ \Rightarrow A_{10} &= -i\frac{g}{\omega} B \end{aligned}$$

Whilst the kinematic condition gives us:

$$\begin{aligned} \frac{\partial A_{10}}{\partial z} + i\omega B &= 0 && @ \quad z = 0 \\ \Rightarrow A_{10} &= -i\frac{\omega}{k} B \end{aligned}$$

For both of these expressions to be correct we need to introduce another condition:

$$\omega^2 = gk \tag{1.V}$$

Which is the linear dispersion relation on infinite depth.

1.2.2 $\mathcal{O}(\varepsilon^2)$

When moving on to the next ε order we insert the results that we acquired to remove all A_{10} terms from our equations. We will also use the dispertion relation (1.V) to simplify the expressions.

We start again with the Laplace equation and beginning with the 2nd harmonic we have a differential equation that is almost identical to the one we solved for A_{10}:

$$4k^2 A_{20} = \frac{\partial^2 A_{20}}{\partial z^2} \qquad \Rightarrow \qquad A_{20} = A_{20}^{+} e^{2kz} + A_{20}^{-} e^{-2kz}$$

We can see that the 1st harmonic has the same differential equation on the left-hand side but there is now a non-homogenous term on the right-hand side so the solution will also get a particular part:

$$\begin{aligned} \frac{\partial^2 A_{11}}{\partial z^2} - k^2 A_{11} &= -2\frac{\omega}{k}\left(k_x \frac{\partial B}{\partial x_1} + k_y \frac{\partial B}{\partial y_1}\right) e^{kz} \\ \Rightarrow A_{11} &= A_{11}^{+} e^{kz} + A_{11}^{-} e^{-kz} - \frac{\omega}{k^2}\left(k_x \frac{\partial B}{\partial x_1} + k_y \frac{\partial B}{\partial y_1}\right) z e^{kz} \end{aligned}$$

The bottom condition removes a term in the homogenous part of the solution for both the 2nd harmonic function:

$$\lim_{z \to -\infty} \frac{\partial A_{20}}{\partial z} = 0 \qquad \Rightarrow \qquad A_{20} = A_{20}^{+} e^{2kz}$$

And the 1st harmonic function:

$$\lim_{z \to -\infty} \frac{\partial A_{11}}{\partial z} = 0 \quad \Rightarrow \quad A_{11} = A_{11}^{+} e^{kz} - \frac{\omega}{k^2}\left(k_x \frac{\partial B}{\partial x_1} + k_y \frac{\partial B}{\partial y_1}\right) z e^{kz}$$

With the dynamic surface condition we now get an expression for the 2nd harmonic containing both B and B_{20}:

$$\begin{aligned} -\frac{1}{4} g k B^2 + \frac{1}{2} g B_{20} - i\omega A_{20} &= 0 \qquad @ \quad z = 0 \\ \Rightarrow A_{20} &= \frac{i\omega}{4k}\left(kB^2 - 2B_{20}\right) e^{2kz} \end{aligned} \tag{1.VI}$$

The following expression for the 1st harmonic:

$$\begin{aligned} k A_{11} - \frac{\omega}{2k^2}\left(k_x \frac{\partial B}{\partial x_1} + k_y \frac{\partial B}{\partial y_1}\right) &= 0 \qquad @ \quad z = 0 \\ \Rightarrow A_{11} &= \frac{\omega}{2k^3}\left(k_x \frac{\partial B}{\partial x_1} + k_y \frac{\partial B}{\partial y_1}\right)(1 - 2kz)\, e^{kz} \end{aligned} \tag{1.VII}$$

And we get that there is no set-down of this magnitude:

$$\begin{aligned} & g\bar{\eta}_0 = 0 && @ \quad z = 0 \\ \Rightarrow & \bar{\eta}_0 = 0 \end{aligned}$$

The kinematic surface condition gives us a 2nd harmonic that is quite similar to (1.VI), where the expression within the brackets are equal yet the constant outside is different:

$$\begin{aligned} & i\frac{\partial A_{20}}{\partial z} = 2\omega B_{20} - k\omega B^2 && @ \quad z = 0 \\ \Rightarrow & A_{20} = i\frac{\omega}{2k}\left(kB^2 - 2B_{20}\right)e^{2kz} && \text{(1.VIII)} \end{aligned}$$

Our expression for the 1st harmonic now contains a $\frac{\partial B}{\partial t_1}$ term, which is completely missing in (1.VII):

$$\begin{aligned} & \frac{\partial B}{\partial t_1} - \frac{\partial A_{11}}{\partial z} = 0 && @ \quad z = 0 \\ \Rightarrow & A_{11} = \left(\frac{\omega}{k^3}\left(k_x\frac{\partial B}{\partial x_1} + k_y\frac{\partial B}{\partial y_1}\right)(1 - kz) + \frac{1}{k}\frac{\partial B}{\partial t_1}\right)e^{kz} && \text{(1.IX)} \end{aligned}$$

We still need to make sure that our solutions are consistent. Since the 2nd harmonic expressions (1.VI) and (1.VIII) contain B_{20} we can use it as a free variable and we then get the expressions:

$$B_{20} = \frac{k}{2}B^2 \qquad \text{and} \qquad A_{20} = 0$$

The 1st harmonic expressions (1.VII) and (1.IX) does not have any free variables and so instead we obtain a condition on B:

$$\frac{\partial B}{\partial t_1} + \frac{\omega}{2k^2}\left(k_x\frac{\partial B}{\partial x_1} + k_y\frac{\partial B}{\partial y_1}\right) = 0 \qquad \text{(1.X)}$$

This is the first part of the evolution equations for the wave envelope. If we apply the the dispertion relation (1.V) to the definition of group velocity we get that:

$$\mathbf{C}_g \overset{\text{def}}{=} \frac{\partial \omega}{\partial \mathbf{k}} = \frac{\partial k}{\partial \mathbf{k}}\frac{\partial \omega}{\partial k} = \frac{\omega}{2k^2}(k_x, k_y)$$

And we now see that the evolution equation (1.X) can also be expressed as:

$$\frac{\partial B}{\partial t_1} + \mathbf{C}_g \cdot \nabla_1 B = 0$$

And in this form we can see that it says that the amplitude moves at the group velocity.

1.2.3 $\mathcal{O}(\varepsilon^3)$

Beginning once again with the Laplace equation we can easily solve both the 3rd harmonic equation:

$$9k^2 A_3 = \frac{\partial^2 A_3}{\partial z^2} \qquad \Rightarrow \qquad A_3 = A_3^+ e^{3kz} + A_3^- e^{-3kz}$$

And the 2nd harmonic equation:

$$4k^2 A_{21} = \frac{\partial^2 A_{21}}{\partial z^2} \qquad \Rightarrow \qquad A_{21} = A_{21}^+ e^{2kz} + A_{21}^- e^{-2kz}$$

While the particular part of the 1st harmonic has become rather messy:

$$\begin{aligned}
\frac{\partial^2 A_{12}}{\partial z^2} - k^2 A_{12} =& i\omega \left[\left(\frac{1}{k} - \frac{k_x^2}{k^3} + \frac{2k_x^2 z}{k^2} \right) \frac{\partial^2 B}{\partial x_1^2} \right. \\
&+ \left(\frac{2k_y^2 z}{k^2} - \frac{k_y^2}{k^3} + \frac{1}{k} \right) \frac{\partial^2 B}{\partial y_1^2} \\
&\left. + \left(\frac{4zk_x k_y}{k^2} - \frac{2k_x k_y}{k^3} \right) \frac{\partial^2 B}{\partial x_1 \partial y_1} \right] e^{kz} \\
\Rightarrow A_{12} =& A_{12}^+ e^{kz} + A_{12}^- e^{-kz} \\
&+ i\omega \left[\left(\frac{z}{2k^2} - \frac{zk_x^2}{k^4} + \frac{k_x^2 z^2}{2k^3} \right) \frac{\partial^2 B}{\partial x_1^2} \right. \\
&+ \left(\frac{k_y^2 z^2}{2k^3} - \frac{zk_y^2}{k^4} + \frac{z}{2k^2} \right) \frac{\partial^2 B}{\partial y_1^2} \\
&\left. + \left(\frac{z^2 k_x k_y}{k^3} - \frac{2zk_x k_y}{k^4} \right) \frac{\partial^2 B}{\partial x_1 \partial y_1} \right] e^{kz}
\end{aligned}$$

With the bottom condition we once again remove a term of the homogenous part of the 3rd harmonic:

$$\lim_{z \to -\infty} \frac{\partial A_3}{\partial z} = 0 \qquad \Rightarrow \qquad A_3 = A_3^+ e^{3kz}$$

2nd harmonic:

$$\lim_{z \to -\infty} \frac{\partial A_{21}}{\partial z} = 0 \qquad \Rightarrow \qquad A_{21} = A_{21}^+ e^{2kz}$$

And 1st harmonic solution:

$$\begin{aligned}
&\lim_{z\to-\infty}\frac{\partial A_{12}}{\partial z}=0\\
\Rightarrow A_{12}=&A_{12}^{+}e^{kz}\\
&+i\omega\left[\left(\frac{z}{2k^2}-\frac{zk_x^2}{k^4}+\frac{k_x^2z^2}{2k^3}\right)\frac{\partial^2 B}{\partial x_1^2}\right.\\
&+\left(\frac{k_y^2z^2}{2k^3}-\frac{zk_y^2}{k^4}+\frac{z}{2k^2}\right)\frac{\partial^2 B}{\partial y_1^2}\\
&\left.+\left(\frac{z^2k_xk_y}{k^3}-\frac{2zk_xk_y}{k^4}\right)\frac{\partial^2 B}{\partial x_1\partial y_1}\right]e^{kz}
\end{aligned}$$

The zeroth harmonic expression doesn't give us any information on the form of the slow drift, instead the condition becomes a part of the evolution equations.

$$\lim_{z\to-\infty}\frac{\partial\bar{\phi}_0}{\partial z_1}=0 \tag{1.XI}$$

When applying the dynamical surface condition we can see that both the 3rd harmonic:

$$\begin{aligned}
&gB_3-3iA_3\omega-\frac{3}{8}k^2gB^3=0 \qquad @\quad z=0\\
\Rightarrow&A_3=i\frac{\omega}{24k}\left(3k^2B^3-8B_3\right)
\end{aligned} \tag{1.XII}$$

And the 2nd harmonic expressions have free variables:

$$\begin{aligned}
&i\frac{g}{4k}B\left(k_x\frac{\partial B}{\partial x_1}+k_y\frac{\partial B}{\partial y_1}\right)+\frac{1}{2}gB_{21}-iA_{21}\omega=0 \qquad @\quad z=0\\
\Rightarrow&A_{21}=\frac{\omega}{4k^2}B\left(k_x\frac{\partial B}{\partial x_1}+k_y\frac{\partial B}{\partial y_1}-2ikB_{21}\right)
\end{aligned} \tag{1.XIII}$$

Which the 1st harmonic expression does not:

$$\begin{aligned}
\frac{3}{16}k^2B\left|B\right|^2g-\frac{1}{2}iA_{12}\omega-\frac{1}{4}\frac{k_xk_yg}{k^4}\frac{\partial^2 B}{\partial x_1\partial y_1}&\\
-\frac{1}{8}\frac{k_y^2g}{k^4}\frac{\partial^2 B}{\partial y_1^2}-\frac{1}{8}\frac{k_x^2g}{k^4}\frac{\partial^2 B}{\partial x_1^2}=0 \qquad @\quad z=0&
\end{aligned}$$

$$\begin{aligned}\Rightarrow A_{12} =& i\omega \left[-\frac{3}{8} kB \left|B\right|^2 \right. \\ &+ \left(\frac{\frac{1}{2}z}{k^2} - \frac{zk_x^2}{k^4} + \frac{\frac{1}{4}k_x^2}{k^5} + \frac{\frac{1}{2}z^2k_x^2}{k^3} \right) \frac{\partial^2 B}{\partial x_1^2} \\ &+ \left(\frac{\frac{1}{2}k_xk_y}{k^5} + \frac{z^2k_xk_y}{k^3} - \frac{2zk_xk_y}{k^4} \right) \frac{\partial^2 B}{\partial x_1 \partial y_1} \\ &+ \left. \left(\frac{\frac{1}{2}z^2k_y^2}{k^3} + \frac{\frac{1}{2}z}{k^2} + \frac{\frac{1}{4}k_y^2}{k^5} - \frac{zk_y^2}{k^4} \right) \frac{\partial^2 B}{\partial y_1^2} \right] e^{kz} \end{aligned} \tag{1.XIV}$$

And we can see that the set down is only dependent on the slow drift:

$$\begin{aligned} \frac{\partial \bar{\phi}_0}{\partial t_1} + g\bar{\eta}_1 = 0 \qquad & @ \quad z = 0 \\ \Rightarrow \bar{\eta}_1 = -\frac{1}{g}\frac{\partial \bar{\phi}_0}{\partial t_1} & \end{aligned} \tag{1.XV}$$

The kinematic surface condition gives us the same content within the brackets as previously for both the 3rd harmonic:

3rd harmonic

$$\begin{aligned} i\frac{9}{16}\omega B^3k^2 - i\frac{3}{2}B_3\omega - \frac{1}{2}\frac{\partial A_3}{\partial z} = 0 \qquad & @ \quad z = 0 \\ \Rightarrow A_3 = i\frac{\omega}{8k}\left(3k^2B^3 - 8B_3\right)e^{3kz} & \end{aligned} \tag{1.XVI}$$

And the 2nd harmonic:

$$\begin{aligned} \frac{\omega k_x}{2k}B\frac{\partial B}{\partial x_1} - iB_{21}\omega - \frac{1}{2}\frac{\partial A_{21}}{\partial z} + \frac{\omega k_y}{2k}B\frac{\partial B}{\partial y_1} = 0 \qquad & @ \quad z = 0 \\ \Rightarrow A_{21} = \frac{\omega}{2k^2}\left(Bk_x\frac{\partial B}{\partial x_1} + Bk_y\frac{\partial B}{\partial y_1} - 2ikB_{21}\right)e^{2kz} & \end{aligned} \tag{1.XVII}$$

The 1st harmonic expression is similar to what we previously found but the coefficients are different:

$$-i\frac{1}{32}\frac{\partial A_{12}}{\partial z}k^2B\left|B\right|^2\omega + i\frac{3}{8}k^2\omega B\left|B\right|^2 = 0 \qquad @ \quad z = 0$$

$$\begin{aligned}\Rightarrow A_{12} =& i\omega \left[\frac{5}{8} kB \left|B\right|^2 \right. \\ &+ \left(\frac{2k_x k_y}{k^5} + \frac{z^2 k_x k_y}{k^3} - \frac{2z k_x k_y}{k^4} \right) \frac{\partial^2 B}{\partial x_1 \partial y_1} \\ &+ \left(\frac{\frac{1}{2}z}{k^2} - \frac{\frac{1}{2}}{k^3} + \frac{k_x^2}{k^5} - \frac{z k_x^2}{k^4} + \frac{\frac{1}{2} z^2 k_x^2}{k^3} \right) \frac{\partial^2 B}{\partial {x_1}^2} \\ &+ \left. \left(\frac{-z k_y^2}{k^4} - \frac{\frac{1}{2}}{k^3} + \frac{k_y^2}{k^5} + \frac{\frac{1}{2}z}{k^2} + \frac{\frac{1}{2} z^2 k_y^2}{k^3} \right) \frac{\partial^2 B}{\partial {y_1}^2} \right] e^{kz} \end{aligned} \tag{1.XVIII}$$

The 0^{th} harmonic expression gives another addition to the evolution equations:

$$\frac{\partial \bar{\phi}_0}{\partial z_1} = \frac{\omega}{2k} \left(k_x \frac{\partial \left|B\right|^2}{\partial x_1} + k_y \frac{\partial \left|B\right|^2}{\partial y_1} \right) \qquad @ \quad z = 0 \tag{1.XIX}$$

Once again we combine the results from our two surface conditions, and by combining the 3^{rd} harmonic expressions (1.XII) and (1.XVI) we get:

$$B_3 = \frac{3k^2}{8} B^3 \qquad \text{and} \qquad A_3 = 0$$

The 2^{nd} harmonic expressions give us:

$$B_{21} = -\frac{i}{2k} B \left(k_x \frac{\partial B}{\partial x_1} + k_y \frac{\partial B}{\partial y_1} \right) \qquad \text{and} \qquad A_{21} = 0$$

Combining the 1^{st} harmonic expressions (1.XIV) and (1.XVIII) gives us the next order terms of the evolution equation:

$$\begin{aligned} i\frac{\omega}{8k^4} \left(\left(3k_x^2 - 2k^2\right) \frac{\partial^2 B}{\partial x_1^2} + \left(3k_y^2 - 2k^2\right) \frac{\partial^2 B}{\partial y_1^2} \right. \\ \left. + 6k_x k_y \frac{\partial^2 B}{\partial x_1 \partial y_1} + 4k^6 B \left|B\right|^2 \right) = 0 \end{aligned} \tag{1.XX}$$

Where special care has been taken to make sure that the units and coefficients are comparable to (1.X)

Finally, by taking the leading order evolution equation (1.X) multiplying by B^* and adding the complex conjugate we get:

$$\frac{\partial \left|B\right|^2}{\partial t_1} + \mathbf{C}_g \cdot \nabla_1 \left|B\right|^2 = 0$$

To satisfy this equation $|B|^2$ must be of the form $f(\mathbf{r}_1 \cdot \mathbf{k} - \frac{1}{2}\omega t_1)$ for some function f. Inserting this into equation (1.XIX) gives us that also $\bar{\phi}_0$ must be of this form and hence it must satisfy the equation:

$$\frac{\partial \bar{\phi}_0}{\partial t_1} + \mathbf{C}_g \cdot \nabla_1 \bar{\phi}_0 = 0$$

Combining this result with equation (1.XV) we can now express the set-down as a spatial derivative of the slow drift:

$$\bar{\eta}_1 = \frac{1}{2\omega k}\left(k_x \frac{\partial \bar{\phi}_0}{\partial x_1} + k_y \frac{\partial \bar{\phi}_0}{\partial y_1}\right)$$

1.3 Evolution equations

We've now acquired the equations (1.X) for order $\mathcal{O}\left(\varepsilon^2\right)$ and (1.XX) for $\mathcal{O}\left(\varepsilon^3\right)$ and by introducing A_{13} in the perturbation expansion we can easily get the fourth order equation. Combining these will then give an accurate description of the evolution of the amplitude B:

$$\begin{aligned}
&\frac{\partial B}{\partial t_1} + \frac{\omega}{2k^2}\left(k_x \frac{\partial B}{\partial x_1} + k_y \frac{\partial B}{\partial y_1}\right) + i\frac{\omega}{8k^4}\left(4k^6 B\left|B\right|^2 + \left(3k_x^2 - 2k^2\right)\frac{\partial^2 B}{\partial x_1^2}\right. \\
&\quad \left. + \left(3k_y^2 - 2k^2\right)\frac{\partial^2 B}{\partial y_1^2} + 6k_x k_y \frac{\partial^2 B}{\partial x_1 \partial y_1}\right) + \frac{k_x \omega}{16k^6}\left(6k^2 - 7k_x^2\right)\frac{\partial^3 B}{\partial x_1^3} \\
&\quad + \frac{k_y \omega}{16k^6}\left(6k^2 - 7k_y^2\right)\frac{\partial^3 B}{\partial y_1^3} + \frac{\omega}{4}B\left(k_x \frac{\partial \left|B\right|^2}{\partial x_1} + k_y \frac{\partial \left|B\right|^2}{\partial y_1}\right) \\
&\quad + \frac{k_y \omega}{16k^6}\left(6k^2 - 21k_x^2\right)\frac{\partial^3 B}{\partial x_1^2 \partial y_1} + \frac{k_x \omega}{16k^6}\left(6k^2 - 21k_y^2\right)\frac{\partial^3 B}{\partial x_1 \partial y_1^2} \\
&\quad + \frac{5\omega}{4}\left|B\right|^2\left(k_x \frac{\partial B}{\partial x_1} + k_y \frac{\partial B}{\partial y_1}\right) + iB\left(k_x \frac{\partial \bar{\phi}_0}{\partial x_1} + k_y \frac{\partial \bar{\phi}_0}{\partial y_1}\right) = 0 \quad \text{(1.XXI)}
\end{aligned}$$

The equations (1.XIX) and (1.XI) describe the limits of the slow drift and in the fourth order Schrödinger equations the Laplace equation gives us a description of it's form. The complete system of fourth order evolution equations for the slow drift are therefore:

$$\begin{aligned}
\frac{\partial^2 \bar{\phi}_0}{\partial x_1^2} + \frac{\partial^2 \bar{\phi}_0}{\partial y_1^2} + \frac{\partial^2 \bar{\phi}_0}{\partial z_1^2} &= 0 \\
\frac{\partial \bar{\phi}_0}{\partial z_1} = \frac{\omega}{2k}\left(k_x \frac{\partial \left|B\right|^2}{\partial x_1} + k_y \frac{\partial \left|B\right|^2}{\partial y_1}\right) \qquad &@ \quad z = 0 \\
\lim_{z \to -\infty} \frac{\partial \bar{\phi}_0}{\partial z_1} &= 0
\end{aligned}$$

1.4 Reconstruction formulas

We have now acquired the complete set of spatially derived reconstruction formulas with respect to B:

$$
\begin{aligned}
A_{10} =& -i\frac{\omega}{k}Be^{kz} \\
A_{11} =& \frac{\omega}{2k^2}\left(\frac{1}{k}-2z\right)\left(k_x\frac{\partial B}{\partial x_1}+k_y\frac{\partial B}{\partial y_1}\right)e^{kz} \\
A_{12} =& i\omega\left[-\frac{3}{8}kB\left|B\right|^2\right. \\
&+\left(\frac{\frac{1}{2}z}{k^2}-\frac{zk_x^2}{k^4}+\frac{\frac{1}{4}k_x^2}{k^5}+\frac{\frac{1}{2}z^2k_x^2}{k^3}\right)\frac{\partial^2 B}{\partial x_1^2} \\
&+\left(\frac{\frac{1}{2}k_xk_y}{k^5}+\frac{z^2k_xk_y}{k^3}-\frac{2zk_xk_y}{k^4}\right)\frac{\partial^2 B}{\partial x_1\partial y_1} \\
&+\left.\left(\frac{\frac{1}{2}z^2k_y^2}{k^3}+\frac{\frac{1}{2}z}{k^2}+\frac{\frac{1}{4}k_y^2}{k^5}-\frac{zk_y^2}{k^4}\right)\frac{\partial^2 B}{\partial y_1^2}\right]e^{kz} \\
A_{20} =& 0 \\
A_{21} =& 0 \\
A_3 =& 0 \\
\bar{\phi}_0 =& \frac{\omega}{2k}\left(k_x\frac{\partial\left|B\right|^2}{\partial x_1}+k_y\frac{\partial\left|B\right|^2}{\partial y_1}\right)e^{kz_1} \\
\bar{\phi}_1 =& \bar{\phi}_1(x_1,y_1,z_1,t_1) \\
B_{20} =& \frac{k}{2}B^2 \\
B_{21} =& -\frac{i}{2k}B\left(k_x\frac{\partial B}{\partial x_1}+k_y\frac{\partial B}{\partial y_1}\right) \\
B_3 =& \frac{3k^2}{8}B^3 \\
\bar{\eta}_0 =& 0 \\
\bar{\eta}_1 =& \frac{1}{2\omega k}\left(k_x\frac{\partial\bar{\phi}_0}{\partial x_1}+k_y\frac{\partial\bar{\phi}_0}{\partial y_1}\right)
\end{aligned}
$$

Chapter 2

Extracting the initial state from a surface data set

In this chapter I will demonstrate how we can extract the characteristic wave vector $\mathbf{k}_c$ and the leading order, first harmonic amplitude B needed to initialize the Schrödinger equations derived in the previous chapter.

2.1 The Fourier transform

Assume we have a harmonic wave with a single constant wavevector $\mathbf{k}_0$ and angular velocity ω_0:

$$\eta(\mathbf{r},t) = \frac{1}{2}\left(Be^{i(\mathbf{k}_0\cdot\mathbf{r}-\omega_0 t)} + B^* e^{-i(\mathbf{k}_0\cdot\mathbf{r}-\omega_0 t)}\right) \tag{2.I}$$

By performing the 2-dimentional spatial Fourier transform along the surface coordinates $\mathbf{r} = (x, y)$ we get:

$$\begin{aligned}
\hat{\eta}(\mathbf{k},t) &= \frac{1}{(2\pi)^2}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\frac{1}{2}\left(Be^{i(\mathbf{k}_0\cdot\mathbf{r}-\omega_0 t)} + B^* e^{-i(\mathbf{k}_0\cdot\mathbf{r}-\omega_0 t)}\right)e^{-i\mathbf{k}\cdot\mathbf{r}}dx\,dy \\
&= \frac{1}{(2\pi)^2}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\frac{1}{2}\left(Be^{i((\mathbf{k}_0-\mathbf{k})\cdot\mathbf{r}-\omega_0 t)} + B^* e^{-i((\mathbf{k}_0+\mathbf{k})\cdot\mathbf{r}-\omega_0 t)}\right)dx\,dy \\
&= \frac{1}{2}\left(B\delta\left(\mathbf{k}_0-\mathbf{k}\right)e^{-i\omega_0 t} + B^*\delta\left(\mathbf{k}_0+\mathbf{k}\right)e^{i\omega_0 t}\right)
\end{aligned} \tag{2.II}$$

Where δ is the Dirac delta defined by:

$$\delta(x) = 0 \mid x \neq 0 \qquad \text{and} \qquad \int_{-\infty}^{\infty}\delta(x)dx = 1$$

And with this result we now have a method to acquire both the wave vector and the amplitude. Since the Dirac function is zero everywhere except where the argument is zero, the only nonzero values on $\tilde{\eta}$ occur when $\mathbf{k} = \mathbf{k}_0$ and $\mathbf{k} = -\mathbf{k}_0$.

Since the two peaks are just the complex conjugates of each other, we won't lose any information by only working with one of them; this symmetry property will hold true as long as the original surface consists of only real numbers, which for an ocean surface is of course always correct. By moving one of the peaks to origo, which can be done by Fourier transforming $\eta e^{-i\mathbf{k}_0 \cdot \mathbf{r}}$, we acquire the leading order, first harmonic amplitude by performing the inverse Fourier transform:

$$B = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} B\delta(\mathbf{k}) e^{i\mathbf{k}\cdot\mathbf{r}} dk_x \, dk_y \tag{2.III}$$

For the cases we will investigate, the amplitude will not be completely constant with regards to $\mathbf{r}$ and, since we will have a discrete B defined on a J-by-L grid, we will need to use the discrete Fourier transform, defined by:

$$B_{j,l} = \frac{1}{JL} \sum_{n=1}^{J} \sum_{m=1}^{L} \hat{B}_{n,m} e^{2i\pi \frac{jn}{J}} e^{2i\pi \frac{lm}{L}}$$

In this form we can clearly see that B satisfies the periodic boundary conditions:

$$B_{j,l} = B_{j+J,l} = Bj, l + L = B_{j+J,l+L}$$

2.2 Failing assumptions

Unfortunelately, the results that were acquired in the previous section were dependent on a number of assumptions that are inevitably incorrect when working with actual ocean surface waves. So this section will deal with the consequences and possible workarounds for our problems.

2.2.1 Wave form assumption

Our first problem is that the wave form assumption (2.I) is flawed. Actual ocean surfaces can not be described by a single constant wave number, a more adequate description is to say that they are superpositions of a vast number of waves with wave numbers that are distributed (hopefully) close to some central value k_0, and that this value remains constant.

One estimate of this value is to create a distribution based on the Fourier transform and let our the characteristic value k_c be the expected value, which for the one-dimentional case and a natural number n would be the formula:

$$k_c = C^{-1} \int_{-\infty}^{\infty} |k| \, |\hat{\eta}|^n \, dk \tag{2.IV}$$

Where C is the normalizing constant defined by:

$$C = \int_{-\infty}^{\infty} |\hat{\eta}|^n \, dk$$

Another method to calculate the value can be to say that k_c is the value that satisfies the equation:

$$\int_{-\infty}^{k_c} |\hat{\eta}|^n \, dk = \int_{k_c}^{\infty} |\hat{\eta}|^n \, dk \tag{2.V}$$

We will investigate how well we can simulate a wave field using these values later.

There are of course events that will make the assumption (2.I) hopelessly inaccurate. One can for example imagine a wave field heading north-west intersecting a similar wave field heading north-east, a naive analysis of the area could then lead to the conclusion that there is a single wave field, heading north. Similar situations can occur with the wave number, our model can handle that there is some variance in the wave numbers but if we try to simulate a surface composed of two or more, equally powerful, wave fields that are distinctly different, we will fail.

2.2.2 Initial surface data

To get the result (2.II) we are dependent on the area being either infinite, or encompassing a whole number of waves. The former is clearly impossible and the latter is highly unlikely to happen by chance. The resulting Fourier transform will therefore not be Dirac delta and would not be so even if the form assumption (2.I) were correct.

This in conjunction with the flawed form assumption means that, to obtain the complex amplitude B, it is no longer satisfactory to just transform the peak, we must instead operate on an area about the peak. Choosing the size of this area is quite hazardous as choosing too small an area means our reconstructed surface will get too low an amplitude, whilst too large an area causes superfluous oscillations to occur of increasing magnitude farther away from the centre of the area.

2.3 The spectral method

After having succesfully extracted the amplitude B from our given surface data we still need to time-evolve it to be able to forecast the ocean surface. To do this we need to solve the evolution equation (1.XXI).

One way of doing this is to use the spectral method; we start by remembering the formula for the inverse Fourier transform of a function $f = f(\mathbf{r}, t)$:

$$f(\mathbf{r}, t) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \hat{f}(\mathbf{k}, t) e^{i\mathbf{k}\cdot\mathbf{r}} dk_x \, dk_y$$

Differentiating this a number of times with regards to x or y

$$\begin{aligned} \Rightarrow \qquad \frac{\partial^{n+m} f(\mathbf{r}, t)}{\partial x^n \partial y^m} &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \hat{f}(\mathbf{k}, t) \frac{\partial^{n+m}}{\partial x^n \partial y^m} e^{i\mathbf{k}\cdot\mathbf{r}} dk_x \, dk_y \\ &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} i^{n+m} k_x^n k_y^m \hat{f}(\mathbf{k}, t) e^{i\mathbf{k}\cdot\mathbf{r}} dk_x \, dk_y \end{aligned}$$

We get that the differentiating operator has been eliminated on the right hand side. Now imagine if we have an partial differential equation like this:

$$\frac{\partial f}{\partial t} = C \frac{\partial f}{\partial x}$$

Replacing the f with its inverse Fourier transform form gives us:

$$\int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \frac{\partial \hat{f}}{\partial t} e^{i\mathbf{k}\cdot\mathbf{r}} dk_x \, dk_y = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} i k_x \hat{f} e^{i\mathbf{k}\cdot\mathbf{r}} dk_x \, dk_y$$

And for these integrals to be equal for all t and $\mathbf{r}$, the arguments also have to be equal:

$$\frac{\partial \hat{f}}{\partial t} = i k_x C \hat{f}$$

The result of using the Fourier transforming the equation is then that it removes all spatial derivatives and we get the analytic solution:

$$\hat{f} = \hat{f}_{t=0} e^{i k_x C t}$$

It is now easy to use this to evaluate the function at any time. Unfortunelately this method doesn't work when the equation has nonlinear terms, if we have a equation like this:

$$\frac{\partial f}{\partial t} = C_1 \frac{\partial f}{\partial x} + C_2 f \frac{\partial f}{\partial x}$$

We can still insert the inverse Fourier transform form:

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\frac{\partial\hat{f}}{\partial t}e^{i\mathbf{k}\cdot\mathbf{r}}dk_x\,dk_y = C_1\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}ik_x\hat{f}e^{i\mathbf{k}\cdot\mathbf{r}}dk_x\,dk_y + C_2\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\hat{f}e^{i\mathbf{k}\cdot\mathbf{r}}dk_x\,dk_y\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}ik_x\hat{f}e^{i\mathbf{k}\cdot\mathbf{r}}dk_x\,dk_y$$

But the method grinds to a halt here, there is no way to remove the integrals and so we do not end up with a nice analytic solution or even an easier equation.

Since the evolution equation for B does have some nonlinear terms we need to choose how to deal with them; one option is to ignore them and solve the equation by the spectral method, another is to keep them and instead utilize a finite difference scheme. Since the spectral method is alot simpler to implement and the influence of the nonlinear terms are limited we will use it to solve the equation.

We start with the linear terms of the evolutions equation (1.XXI):

$$\begin{aligned}&\frac{\partial B}{\partial t_1}+\frac{\omega}{2k^2}\left(k_x\frac{\partial B}{\partial x_1}+k_y\frac{\partial B}{\partial y_1}\right)\\&\quad+i\frac{\omega}{8k^4}\left(\left(3k_x^2-2k^2\right)\frac{\partial^2 B}{\partial x_1^2}+\left(3k_y^2-2k^2\right)\frac{\partial^2 B}{\partial y_1^2}+6k_xk_y\frac{\partial^2 B}{\partial x_1\partial y_1}\right)\\&\qquad+\frac{k_x\omega}{16k^6}\left(6k^2-7k_x^2\right)\frac{\partial^3 B}{\partial x_1^3}+\frac{k_y\omega}{16k^6}\left(6k^2-7k_y^2\right)\frac{\partial^3 B}{\partial y_1^3}\\&\quad+\frac{k_y\omega}{16k^6}\left(6k^2-21k_x^2\right)\frac{\partial^3 B}{\partial x_1^2\partial y_1}+\frac{k_x\omega}{16k^6}\left(6k^2-21k_y^2\right)\frac{\partial^3 B}{\partial x_1\partial y_1^2}=0\end{aligned}$$

For an actual surface, we will not have a single wave vector $\mathbf{k}_0$, instead we will use the characteristic wave vector $\mathbf{k}_c = (k_{cx}, k_{cy})$ Since we multiply the original surface with $e^{-i\mathbf{k}_c\cdot\mathbf{r}}$ before performing the Fourier transform, to transform the equations we simply need to do the replacements:

$$B\to\hat{B}\qquad\text{and}\qquad\frac{\partial}{\partial x_1}\to i\kappa_x\qquad\text{and}\qquad\frac{\partial}{\partial y_1}\to i\kappa_y$$

Where $\kappa_x = k_x - k_{cx}$ and $\kappa_y = k_y - k_{cy}$.

Doing these replacements on our equation then gives us:

$$\frac{\partial \hat{B}}{\partial t_1} + ih(k_x, k_y)\hat{B} = 0$$

Where:

$$\begin{aligned} h(k_x, k_y) = \frac{\omega}{2k^2}\Bigg[& (k_x\kappa_x + k_y\kappa_y) - \frac{k_x}{8k^4}\left(6k^2 - 7k_x^2\right)\kappa_x^3 - \frac{k_y}{8k^4}\left(6k^2 - 7k_y^2\right)\kappa_y^3 \\ & - \frac{1}{4k^2}\left(\left(3k_x^2 - 2k^2\right)\kappa_x^2 + \left(3k_y^2 - 2k^2\right)\kappa_y^2 + 6k_xk_y\kappa_x\kappa_y\right) \\ & - \frac{k_y}{8k^4}\left(6k^2 - 21k_x^2\right)\kappa_x^2\kappa_y - \frac{k_x}{8k^4}\left(6k^2 - 21k_y^2\right)\kappa_x\kappa_y^2\Bigg] \end{aligned}$$

This equation then has the easy analytic solution:

$$\hat{B} = \hat{B}_{t=0}e^{-i\varepsilon h(k_x,k_y)t} \tag{2.VI}$$

2.4 The Crank-Nicolson scheme

If we were to use the Crank-Nicolson scheme it would entail introducing discrete spatial and temporal variables ($x_n = n\Delta x$, $y_m = m\Delta y$ and $t_k = k\Delta t$) and replacing all derivatives with the following difference approximations:

$$
\begin{aligned}
\frac{\partial B(x,y,t)}{\partial t} &\approx \frac{1}{\Delta t}\Big(B(x_n,y_m,t_{k+1}) - B(x_n,y_m,t_k)\Big)\\
\frac{\partial B(x,y,t)}{\partial x} &\approx \frac{1}{4\Delta x}\Big(B(x_{n+1},y_m,t_k) - B(x_{n-1},y_m,t_k)\\
&\qquad +B(x_{n+1},y_m,t_{k+1}) - B(x_{n-1},y_m,t_{k+1})\Big)\\
\frac{\partial B(x,y,t)}{\partial y} &\approx \frac{1}{4\Delta y}\Big(B(x_n,y_{m+1},t_k) - B(x_n,y_{m-1},t_k)\\
&\qquad +B(x_n,y_{m+1},t_{k+1}) - B(x_n,y_{m-1},t_{k+1})\Big)\\
\frac{\partial^2 B(x,y,t)}{\partial x^2} &\approx \frac{1}{2\Delta x^2}\Big(B(x_{n+1},y_m,t_k) - 2B(x_n,y_m,t_k)\\
&\qquad + B(x_{n-1},y_m,t_k) + B(x_{n+1},y_m,t_{k+1})\\
&\qquad -2B(x_n,y_m,t_{k+1}) + B(x_{n-1},y_m,t_{k+1})\Big)\\
\frac{\partial^2 B(x,y,t)}{\partial x\partial y} &\approx \frac{1}{8\Delta x\Delta y}\Big(B(x_{n+1},y_{m+1},t_k) - B(x_{n+1},y_{m-1},t_k)\\
&\qquad - B(x_{n-1},y_{m+1},t_k) - B(x_{n-1},y_{m-1},t_k)\\
&\qquad + B(x_{n+1},y_{m+1},t_{k+1}) - B(x_{n+1},y_{m-1},t_{k+1})\\
&\qquad - B(x_{n-1},y_{m+1},t_{k+1}) - B(x_{n-1},y_{m-1},t_{k+1})\Big)\\
\frac{\partial^2 B(x,y,t)}{\partial y^2} &\approx \frac{1}{2\Delta y^2}\Big(B(x_n,y_{m+1},t_k) - 2B(x_n,y_m,t_k)\\
&\qquad + B(x_n,y_{m-1},t_k) + B(x_n,y_{m+1},t_{k+1})\\
&\qquad -2B(x_n,y_m,t_{k+1}) + B(x_n,y_{m-1},t_{k+1})\Big)
\end{aligned}
$$

We would then get a difference equation in the form of a banded matrix system that we can solve iteratively.

Chapter 3

Recreating the Draupner freak wave

On January 1. 1995 a wave hit the Draupner E platform, located in the central North Sea that was considerably larger than the surrounding wave field should be able to generate. In this chapter we will show how the methods developed in the previous chapters can be used to reconstruct the ocean surface and look at ways to measure the accuracy of the reconstruction.

3.1 Measurements and general sea state

The Draupner platform was equipped with a downwards facing laser-based sensor recording surface elevation at a speed of 2.1333 Hz. This sensor recorded data for 20 minutes every hour and the extreme wave event occurred in the time series that was started at 1520 GMT.

In figure 3.1 we can see that the Draupner wave clearly stands out, having a crest height of 18.5 m, a down-crossing wave height of 25.6 m and an up-crossing wave height of 25.0 m. The surrounding waves had a significant wave height $H_s = \sqrt{8}\bar{a} = 11.9m$. One possible explanation for how such a wave can occur is that multiple waves with different wave numbers, and therefore moving at different velocities, happened to coincide at a point and were 'stacked' on top of each other.

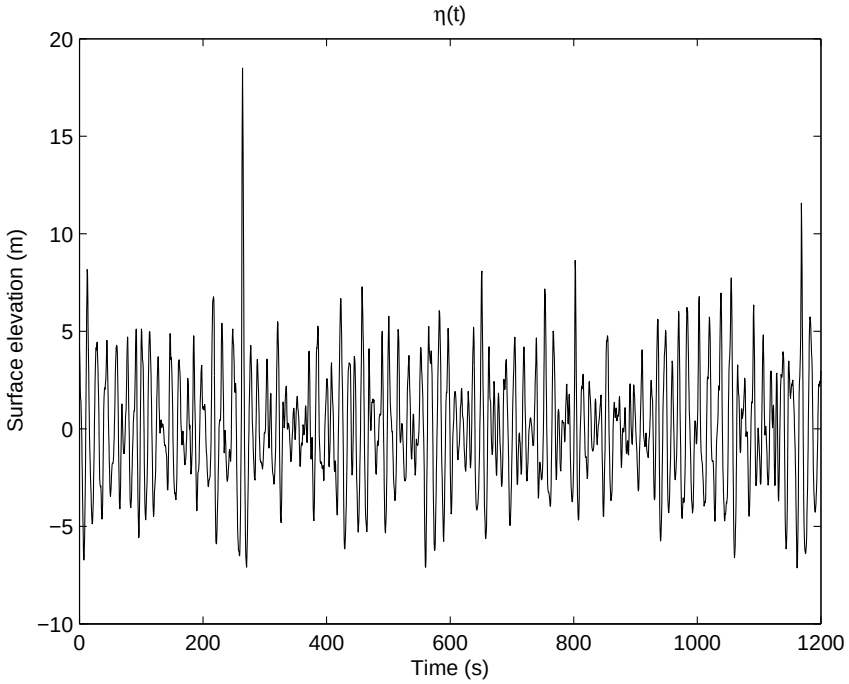

Figure 3.1: Time series showing the Draupner wave

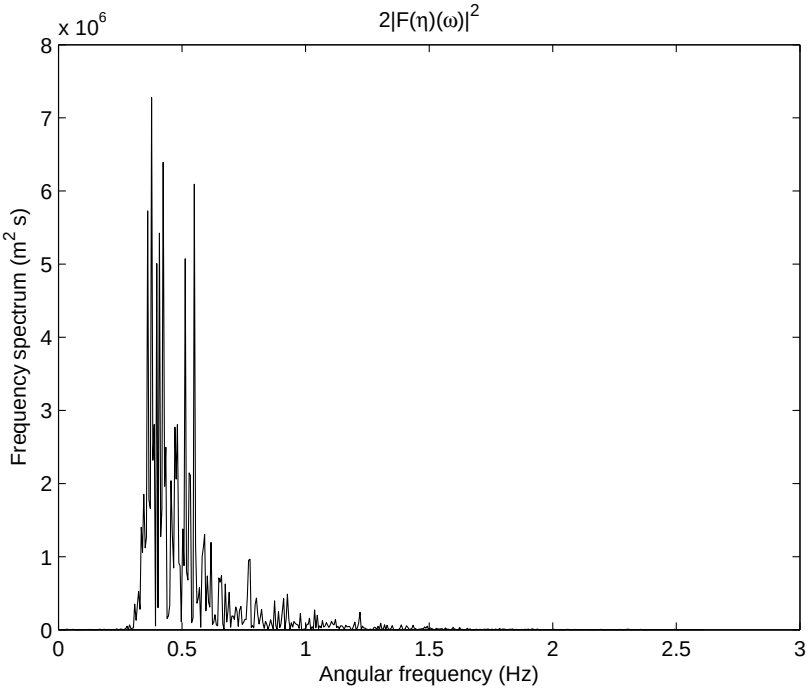

Figure 3.2: Fourier transform of Draupner series

3.2 Recreation

The first thing we need to do is calulate the Fourier transform of the surface data. Using *Matlab* this is easily accomplished by using the *fft* command, and the entire script that has been used to generate all the figures in this chapter can be found in section A.1

3.2.1 Characteristic angular frequency

In figure 3.2 we can see that the Fourier transform of an actual ocean surface does not produce a nice Dirac delta function and so we calculate the expected value with the discrete version of the formula given in chapter 2:

$$\omega_c = C^{-1} \sum_{n=1}^{N} \omega_n \left|\hat{\eta}\right|_n^2 = 0.52Hz \qquad \text{where} \qquad C = \sum_{n=1}^{N} \left|\hat{\eta}\right|_n^2$$

3.2.2 First order, first harmonic amplitude

We can see in figure 3.2 that the angular frequency $\omega_c = 0.52$ corresponds to an area to the right of maximum value. Choosing an area around our characteristic frequency and shifting the area such that 0.52 becomes the origo in the new coordinate system gives us $\hat{B}$, the Fourier transform of the first harmonic amplitude, as seen in figure 3.3.

Using *Matlab*'s inverse Fourier transform function, *ifft*, we get the amplitude B, seen in figure 3.4. And in figure 3.5 we can see how the wave envelope, $\pm|B|$, covers the wave field pretty accurately though it has a tendency to exaggerate the throughs and understate the crests, this is because the higher harmonic amplitudes and the set-down has not yet been taken into account.

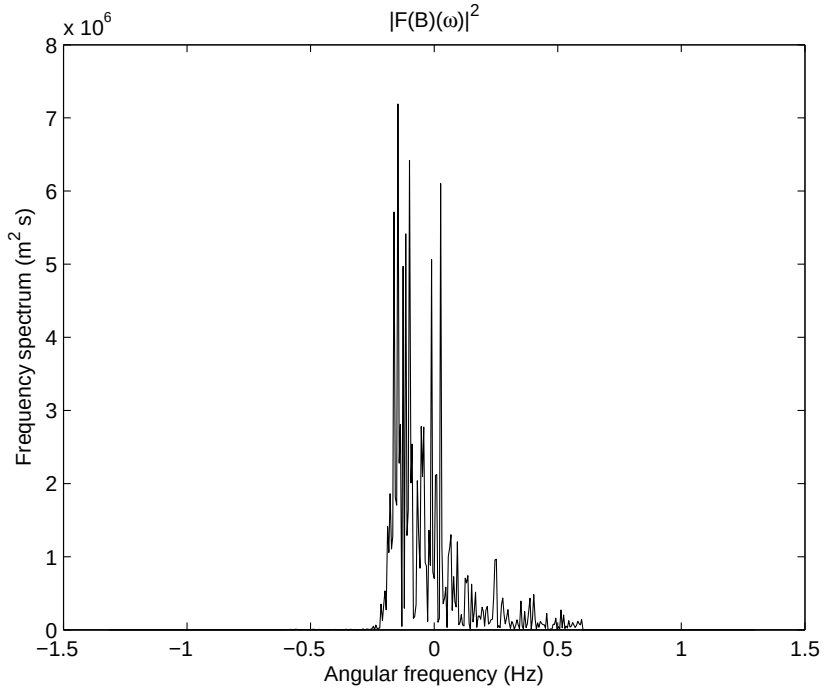

Figure 3.3: Fourier transform of the complex amplitude B

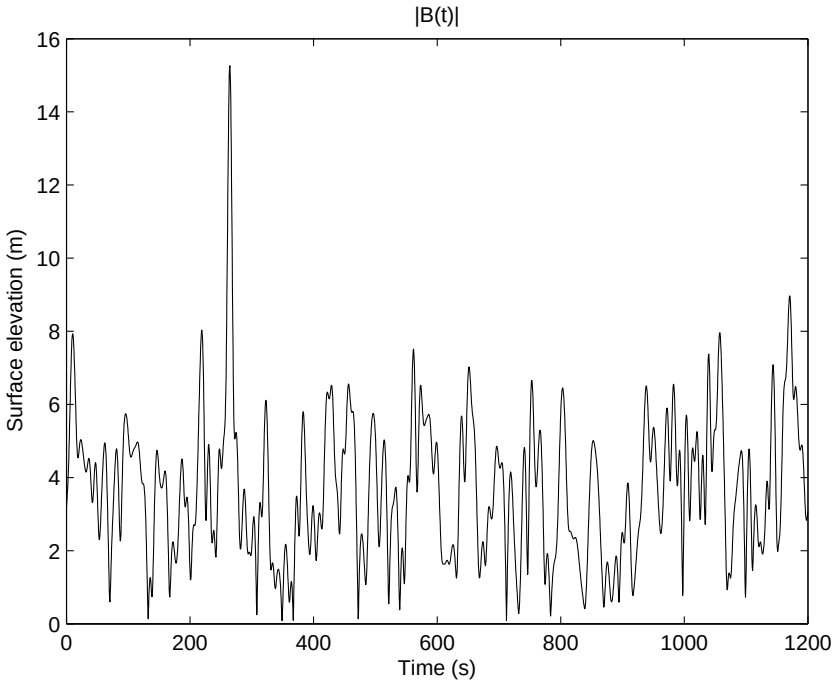

Figure 3.4: B, first order, first harmonic amplitude

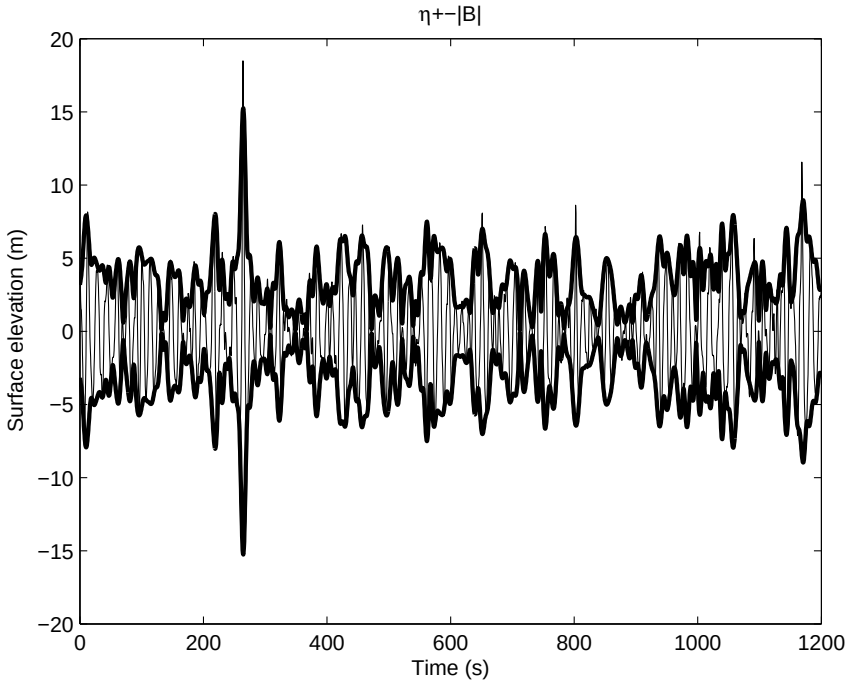

Figure 3.5: The surface and the wave envelope

3.2.3 Higher order amplitudes

Using the reconstruction formulas from section 1.4 we can acquire the higher order amplitudes. By using the evolution equation (1.X) to convert the spatial derivatives to temporal ones we get the reconstruction formulas:

$$\begin{aligned} B_2 =& \frac{k}{2}B^2 + \frac{ik}{\omega}B\frac{\partial B}{\partial t_1} \\ B_3 =& \frac{3k^2}{8}B^3 \\ \bar{\eta} =& 0 \end{aligned}$$

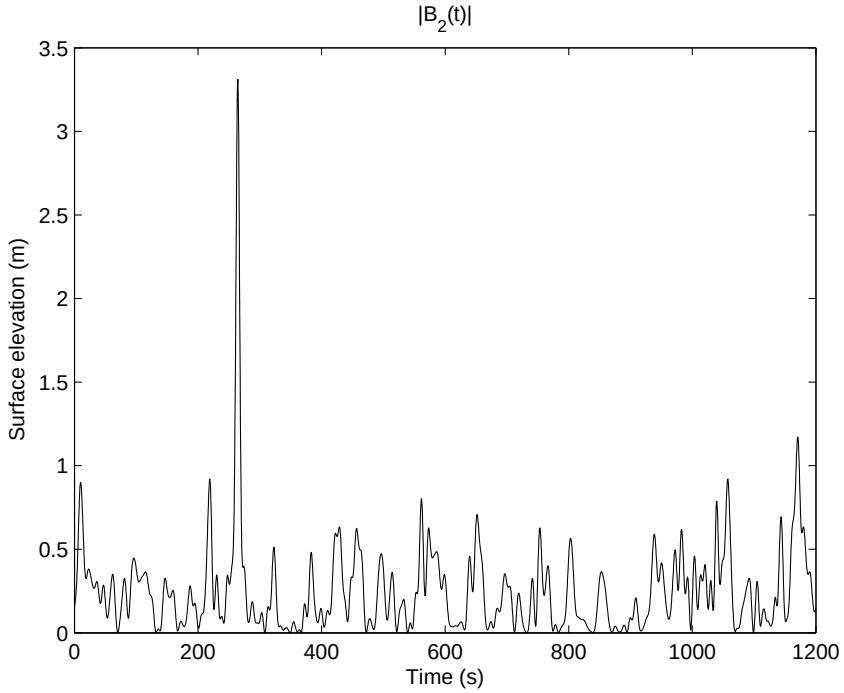

Figure 3.6: The second harmonic amplitude, B_2

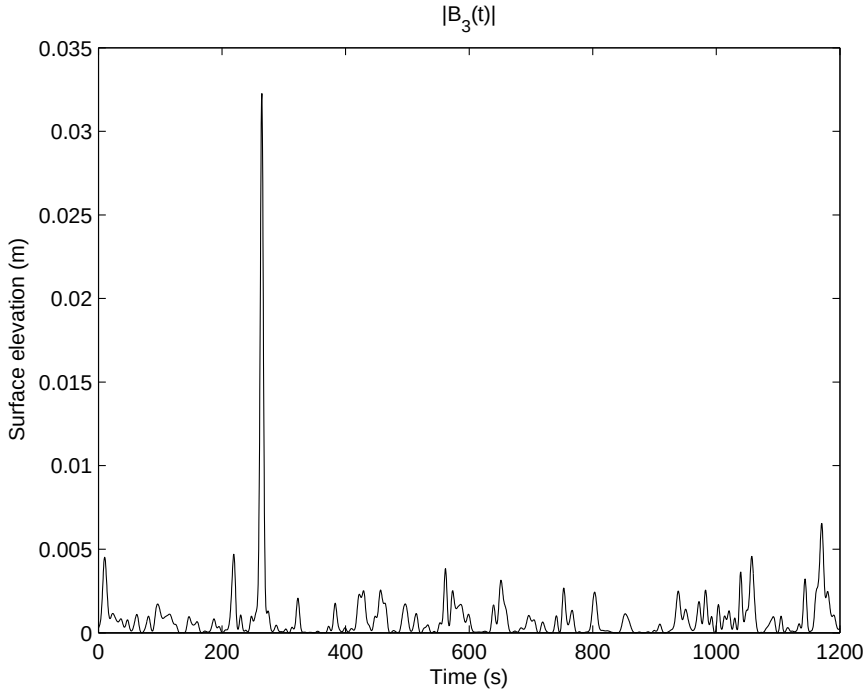

Figure 3.7: The third harmonic amplitude, B_3

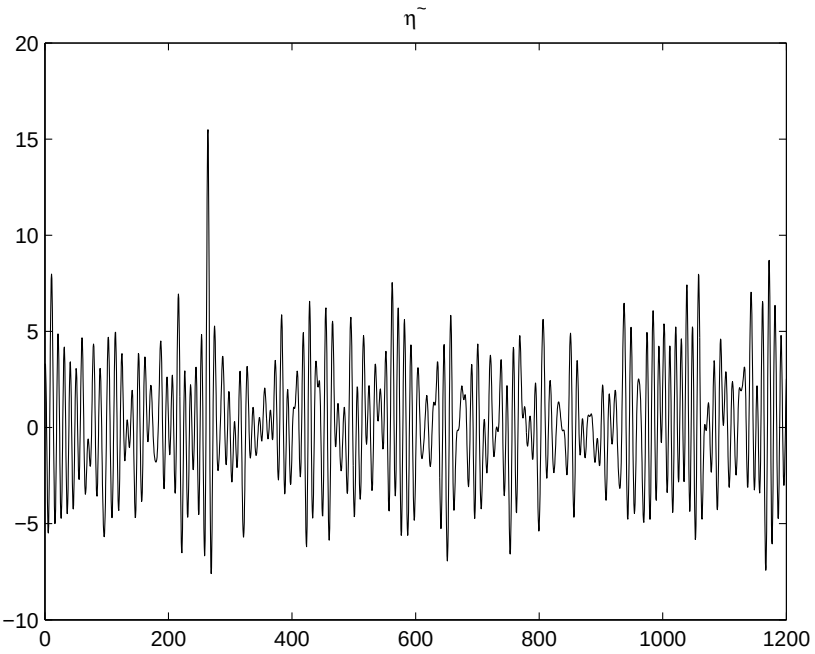

Figure 3.8: The recreated surface

3.3 The recreated surface

Using all the different order amplitudes we can now plot the recreated surface, as seen in figure 3.8. Compared to the original surface in figure 3.1 we can see that we've managed to recreate the actual Draupner wave fairly accurately though we have not been able to recreate the entire time series accurately.

By zooming in on the area around the Draupner wave we can see where some of the problems occur. In figure 3.9 there are some waves that have been recreated almost perfectly, alot that are slightly out of phase, and a few where a crest in the original surface has become a through in the recreated. The recreated surface will, by definition, have a completely constant period, which is not true for the original surface.

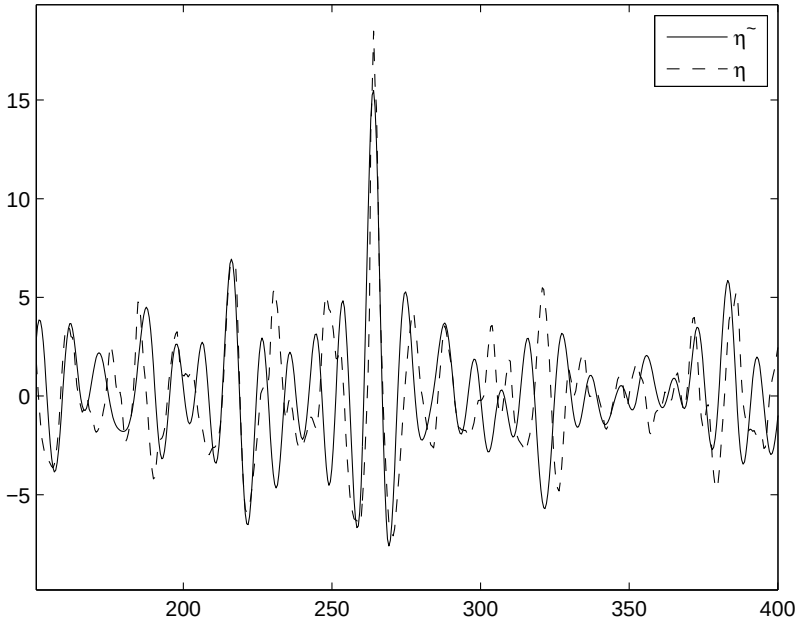

Figure 3.9: A zoomed in view of both the original surface η and the recreated surface $\tilde{\eta}$

Since we only have a point-based measurement it is not possible to use this data to either propagate the surface in time or recreate how the surface is around the measurement point. To do this we would have to have some knowledge of the directional distribution of the waves as in it's current form we have no way of knowing if there are different wave fields converging to our point and from which direction they are coming.

In the next two chapters, we will look at data sets that contain enough information for us to try to propagate the recreated surface.

Chapter 4

Simulating a synthetic surface

In this chapter we will create a simple synthetic surface and then see how well we can replicate the surface with the methods we have developed. The script that is used in this chapter can be found in section A.2.

4.1 The synthetic surface

We want to create a simple harmonic wave field:

$$\eta = Be^{i(\mathbf{k}\cdot\mathbf{r}-\omega t)} + B^* e^{-i(\mathbf{k}\cdot\mathbf{r}-\omega t)}$$

To represent this we have to discretize the area and to be easily comparable we will use the same parameters that will be used in chapter 5. The surface will be represented by a 128-by-128 point grid where the distance between adjacent points is 7.5 metres, the time will be represented by a 32 point array with 1.38 seconds separating each value.

We choose to let B be equal to 1, meaning a 128-by-128 matrix where all elements equals 1, at $t = 0$ and time-evolve it according to the solution (2.VI) to the fourth order linear evolution equation.

Setting the wave vector to $\mathbf{k} = (0.08, 0.06)$ leads to the constants $k = 0.1$, $\omega = 0.99$, $\mathbf{c} = (7.92, 5.94)$ and $\mathbf{C}_g = (3.96, 2.97)$, and a surface like in figure 4.1. The depicted surface is taken at $t = 0$ and we will also create 31 more images taken at a 1.38 second interval for comparison with the recreated surface.

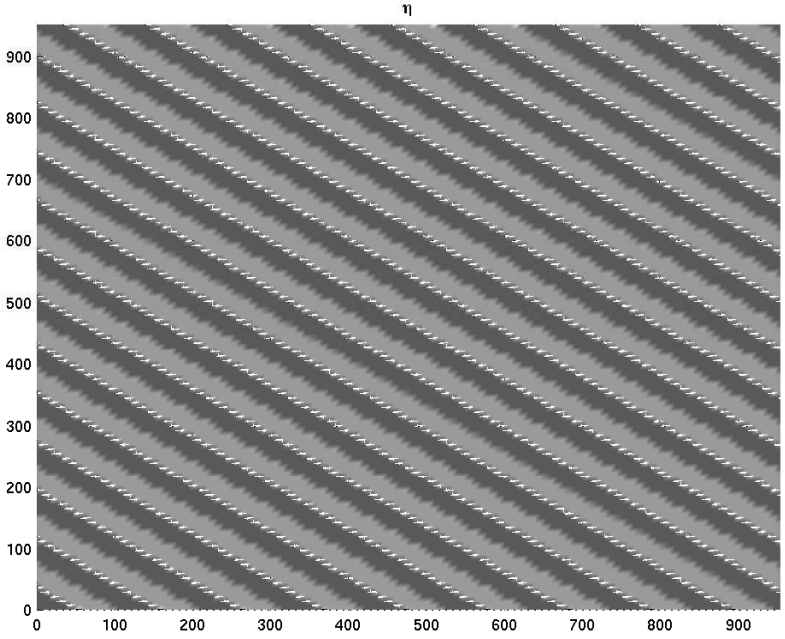

Figure 4.1: Created sea surface at $t = 0$

4.2 Extracting the defining variables

Given only the surface image 4.1 we must now try to recreate both that surface and the subsequent 31 ones with as high accuracy as possible. Note that we are not under any obligation to keep the coordinate system we used to create the surface; we can for example say that the inital image was taken at $t = 10$, this will give us a different B but the surface will still be recreated the same. For convenience we will keep the old system with the initial image taken at $t = 0$.

4.2.1 The Fourier transform

We will once again begin with the Fourier transform. Since we now have a 2-dimentional surface we will utilize the *fft2* function and we get figure 4.2. We can see that it is composed of two clear peaks but even though we created the surface with a single constant wave number, the transform is still not a Dirac function. The two main reasons for this are that the surface does not have a whole number of waves in it and that the wave number does not hit a grid point.

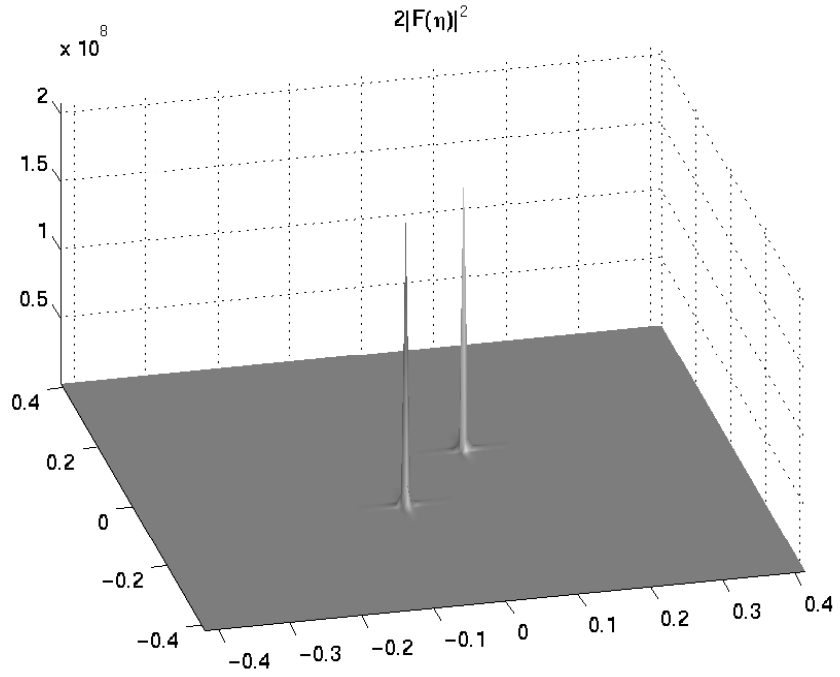

Figure 4.2: Fourier transform of the ocean surface at $t = 0$

4.2.2 Wave vector

We acquire the wave vector by calculating the wave number and the angle between components separately. By integrating the Fourier transform 4.2 with regards to k we get figure 4.4 and by performing an angular integration we get figure 4.3.We can see that both figures are quite close to what one would expect.

As was mentioned in section 2.2.1 there is no method that is proven to produce the optimal characteristic wave vector. To decide which method we will use we'll experiment a bit by calculating the characteristic wave number through what I will refer to as the expectation method, equation (2.IV), and the integration method, equation (2.V), using various values of n.

In figure 4.5 we can see that the estimates with $n = 1$ are completely worthless, the expectation method seems to come quite close to k_0 when $n = 2$ and both methods seem to converge fast to a constant that is a bit less than k_0 when n increases. The value that the expectation method obtains when $n = 2$ is 0.0997 and the value the functions limit is 0.0979.

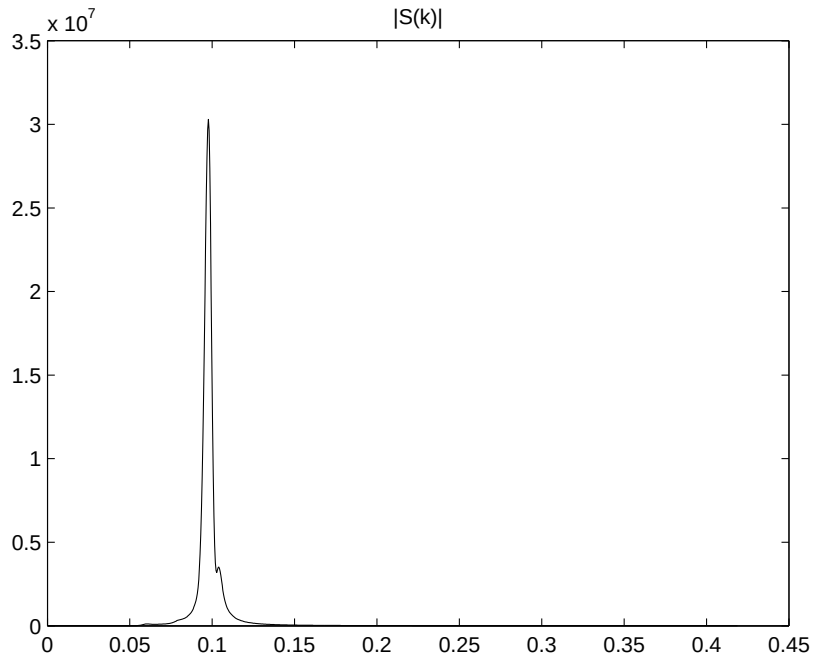

Figure 4.3: Angular integration of the Fourier transform

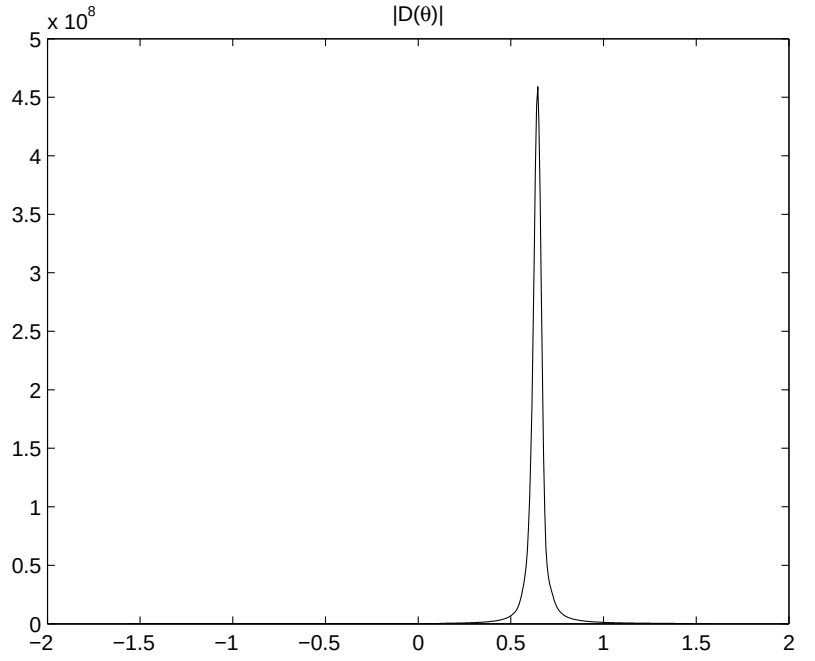

Figure 4.4: Radial integration of the Fourier transform

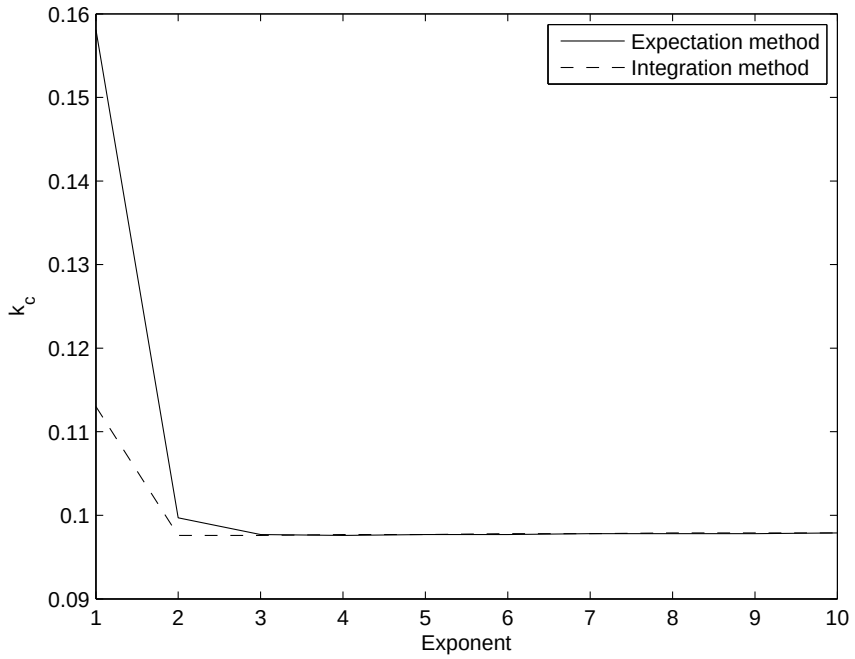

Figure 4.5: The estimate for $k = 0.1$ when using different exponents

I have also performed this test on a surface generated with wave number $k_0 = 0.2$ and the result of that can be seen in figure 4.6. The results are quite similar with the expectation method with $n = 2$ still giving the best estimate, $k_c = 2.008$, though it is now slighty more than k_0 and both functions still seem to be converging to a single limit, $k_c = 0.1959$, but with a considerably slower convergence rate.

Of the methods that we have tried, the most reliable one seems to be the expectation method with exponent $n = 2$ which gave the characteristic wave number $k_c = 0.0997$. A small thing that might be of interest is that the quotient between the original wave number k_0 and the limit was quite close for both $k_0 = 0.1$ and $k_0 = 0.2$, being respectively 1.0215 and 1.0209. There is a possibility that one could develop a method that uses a higher exponent and multiplies it with some constant to obtain a more optimal value for k_c. One of the possible advantages of using a higher exponent is that one reduces the influence of wave numbers that do not appear very frequently. Unfortunelately we have no guarantee that any of the methods used here are of any use when applied on an actual ocean surface.

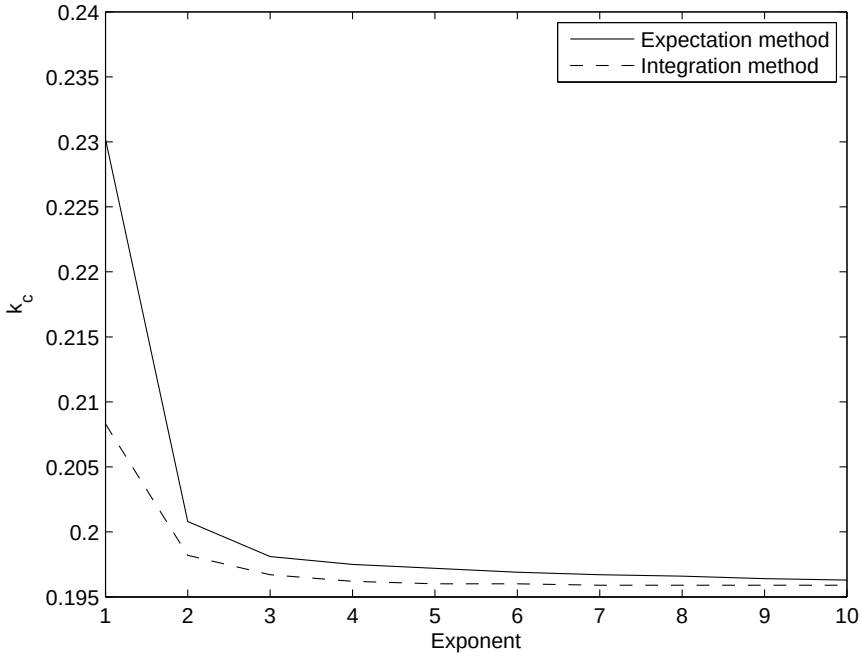

Figure 4.6: The estimate for $k = 0.2$ when using different exponents

Using the expectation method with $n = 2$ on the spectrum and directional distribution then gives us the characteristic wave number $k_c = 0.0997$ and angle $\theta = 0.6331$, this in turn gives us the characteristic wave vector $\mathbf{k}_c = (0.0804, 0.0590)$.

We then get that $\omega_c = 0.9892$, $\mathbf{c} = (8.00, 5.87)$ and $\mathbf{C}_g = (4.00, 2.93)$. These results are quite accurate however this error will limit how long we can use our forecasting method. Since the wave number decides the group speed, any error will lead to the actual waves and the simulation being increasingly out of phase over time and eventually leave the forecasting useless. Therefore it is essential to have both an accurate method of obtaining the characteristic wave number from an actual ocean surfaces and to understand for how long our simulation gives meaningful results.

4.2.3 First harmonic amplitude

By calculating the Fourier transform of $\eta e^{-i\mathbf{r}\cdot\mathbf{k}}$, removing the peak not centered on origo and performing the *ifft2* operation we will get the amplitude B. As we can see in figure 4.7, the B matrix that we obtain, though it's magnitude is close to what we started with, is not exactly what we wanted.

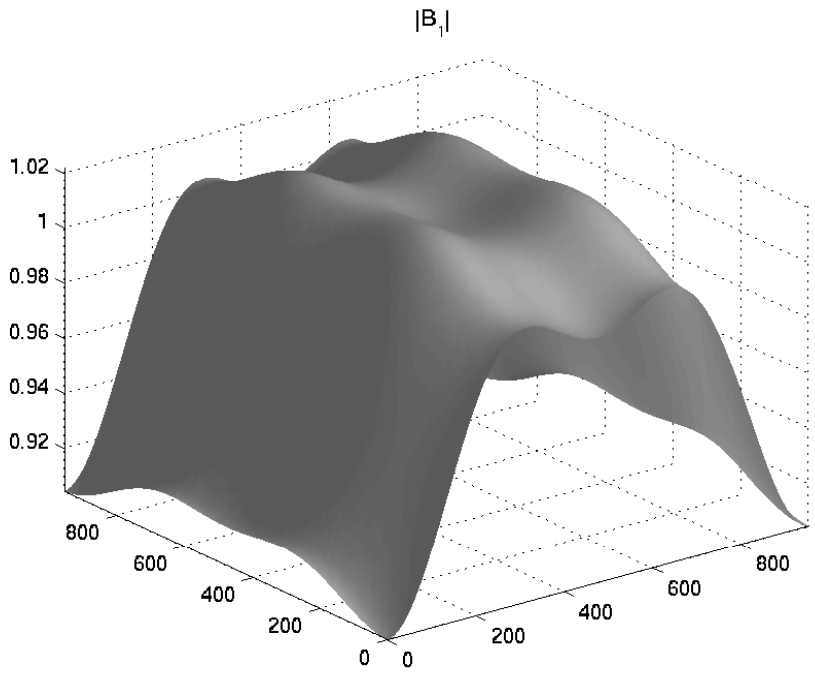

Figure 4.7: First order amplitude at $t = 0$

The main reason why the B matrix has becomes 'curved' is that the wave vector we used was not precise. This caused us to slightly misalign the peak before performing the inverse Fourier transform, we will see later the effect that this error has on our reconstruction. Another reason is that the peak was not a Dirac delta, but for this to occur the original image would have to have had a whole number of waves in it, and the wave vector would have to fall on a grid point. These are not reasonable assumptions.

This amplitude can be improved by applying a low pass filter before performing the inverse Fourier transform. To do this we simply choose some boundary about origo and set everything outside of this boundary equal to zero. Doing this will make the transform closer to a pure Dirac delta and the 'curving' of the surface will be reduced. This procedure does have a price; though the amplitude we get will be closer to constant, this constant will be lower than the one we started with. The best amplitude I have managed to recover is one whose absolute value is close to constantly 0.9564 and it is this amplitude that is used further.

Now that we have the first harmonic amplitude we can easily get the higher order amplitudes by using the recreation formulas from section 1.4 and by using them we get the recreated surface in figure 4.8, which, from a top-down view, is visually indistinguishable from the original surface.

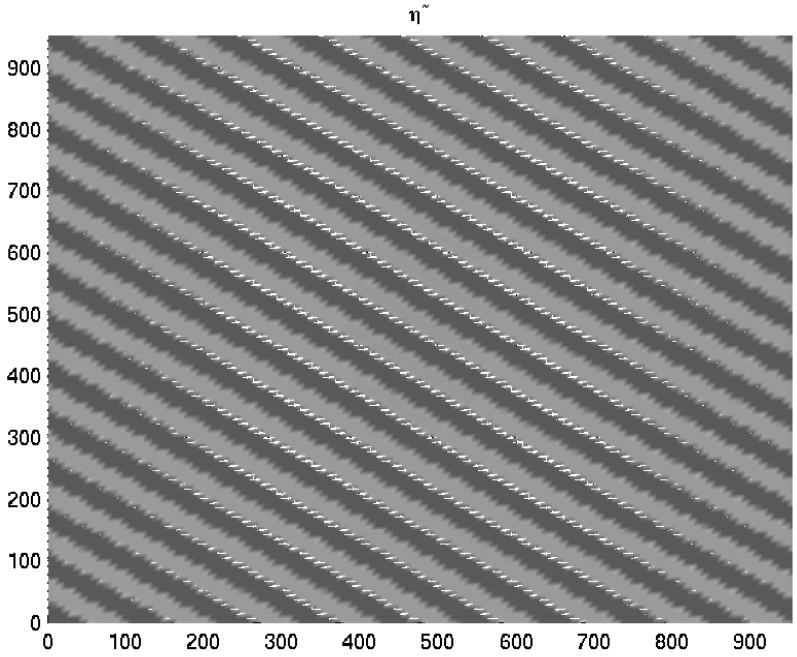

Figure 4.8: Recreated sea surface at $t = 0$

4.3 Time-evolving the surface

We now have all the information and methods necessary to propagate the surface forwards in time. But before we do that we need to develop some methods to measure how accurately we are recreating the surface because as we have seen, the purely visual method is not very helpful.

4.3.1 Error functions based on the L_p norms

The methods we will develop are all based on the L_p norms which, for a discrete 2-dimentional function $f_{k,l}$ defined on a N-by-N grid, is defined as:

$$||f||_p \stackrel{\text{def}}{=} \left(\sum_{k=1}^{N} \sum_{l=1}^{N} |f_{k,l}|^p \right)^{\frac{1}{p}}$$

The norms we will be working with are the L_1, L_2 and L_∞ norms, and in the latter case the definition reduces to $||f||_\infty = \max_{1 \leq k,l \leq N} |f_{k,l}|$.

We can see that, assuming that $f \neq 0$, we can increase the norm by increasing the grid size parameter N. Our error function must, of course, be independent of the grid size and so we need to divide the norm by a normalizing constant C.

We have alot of freedom in choosing this constant so we introduce the condition that if we were to use a completely flat surface as the recreated surface, we should get an error of 1. Then we need only set f equal to the difference between the original surface η and the recreated surface $\tilde{\eta}$ and we can decide the normalizing constant:

$$Err_p(t) = \frac{||\eta(t) - \tilde{\eta}(t)||_p}{C} = \frac{||\eta(t)||_p}{C} = 1 \qquad \Rightarrow \qquad C = ||\eta(t)||_p$$

4.3.2 Decreasing amount of comparable area

One important point that we have to be aware of is that the sequence of radar images that we start with and the sequence of surfaces that we recreate will not cover the same areas. Let's set the coordinate of the lower left corner of the first radar image as (0,0). Then all the subsequent radar images will cover the same area and will also have (0,0) as the lower left corner. The surface images that we create by propagating the first radar images will on the other hand not cover the same area. We are only modeling the waves that are present in the first image and so we are following them as they are moving at group speed, so an image that depicts the surface Δt after the initial radar image will have a lower left corner coordinate of $\mathbf{C}_g \Delta t$.

To compensate for this we will need to calculate which grid points of the original surface have values in the new grid used by the recreated surface and then interpolate one of the grids onto the other before we can compute the error. One consequence that can not be helped is that the area that is comparable will decrease and eventually the two images will be completely separate.

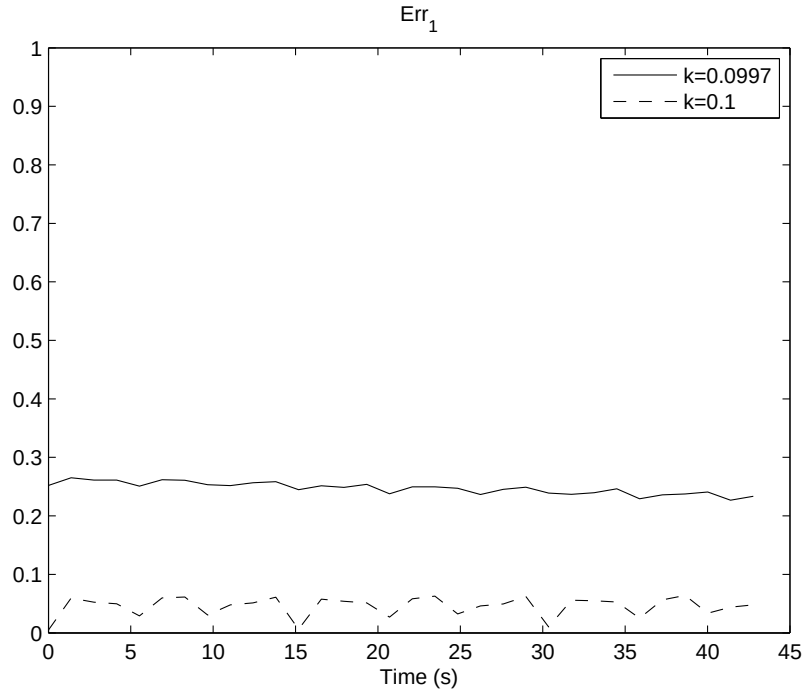

Figure 4.9: Err_1 of the wave propagation using the actual $k = 0.1$ and the approximate $k = 0.0997$

4.3.3 The error plots

We can now begin to investigate the errors in our model. We begin by looking at the 'curved' amplitude B at $t = 0$, in figure 4.7 we saw that it was not exactly equal to 1 and I claimed that the biggest error-source was that we used the wave number equal to 0.0997 instead of 0.1. If we now calculate the different error functions on the B matrix instead of the surface we get the errors $Err_1 = 0.2971$, $Err_2 = 0.3572$ and $Err_\infty = 0.6922$ when using the approximate $k = 0.0997$ and a boundary producing a 'curved' B. With the constant but low magnitude B matrix we get $Err_p = 0.2737$ and using the exact $k = 0.1$, with either boundary area, gives us the error $Err_p = 2.2166 \cdot 10^{-4}$ for all values of p. We can see that there is some numerical error present even with the original wave number but nothing as dramatic as what we get when there is only a slight error in the approximation.

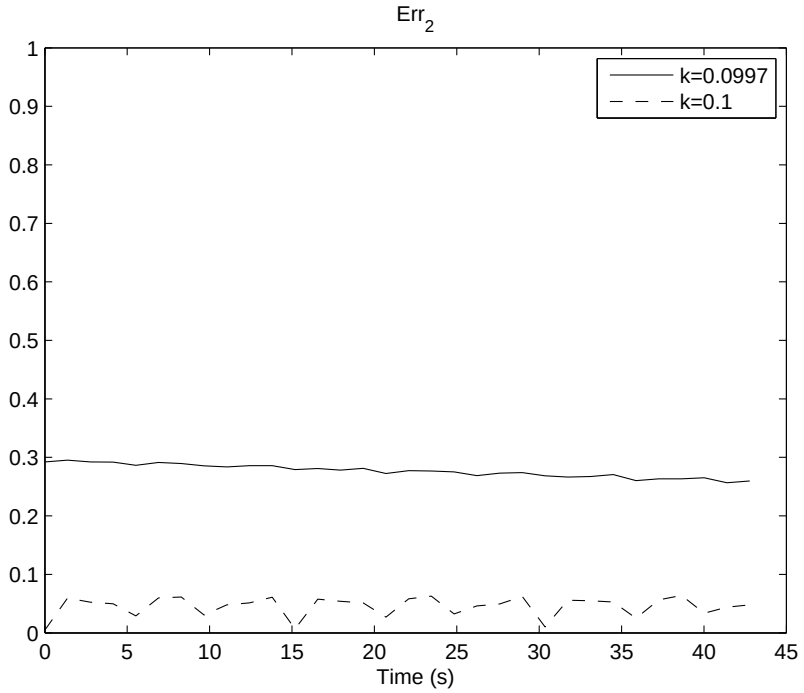

Figure 4.10: Err_2 of the wave propagation using the actual $k = 0.1$ and the approximate $k = 0.0997$

Moving on to the surface, we have the Err_1, Err_2 and Err_∞ plots as the surface is propagated in figures 4.9, 4.10 and 4.11, respectively. We can see that if the surface has been recreated using the actual $k = 0.1$, all the errors are virtually identical and remain firmly below 0.1.

When recreating the surface using the approximate $k = 0.0997$ however, the Err_1, Err_2 and Err_∞ start at around 0.25, 0.3 and 0.625, respectively. This is a dramatic increase in error, considering that the relative error in k is only about 0.003.

One very good thing that we can see in the figures is that the errors seem to be quite stable over time. This could mean that the method we are using to propagate the amplitude and recreate the surface can be quite effective, even if the initial amplitude is not as accurate as we would like.

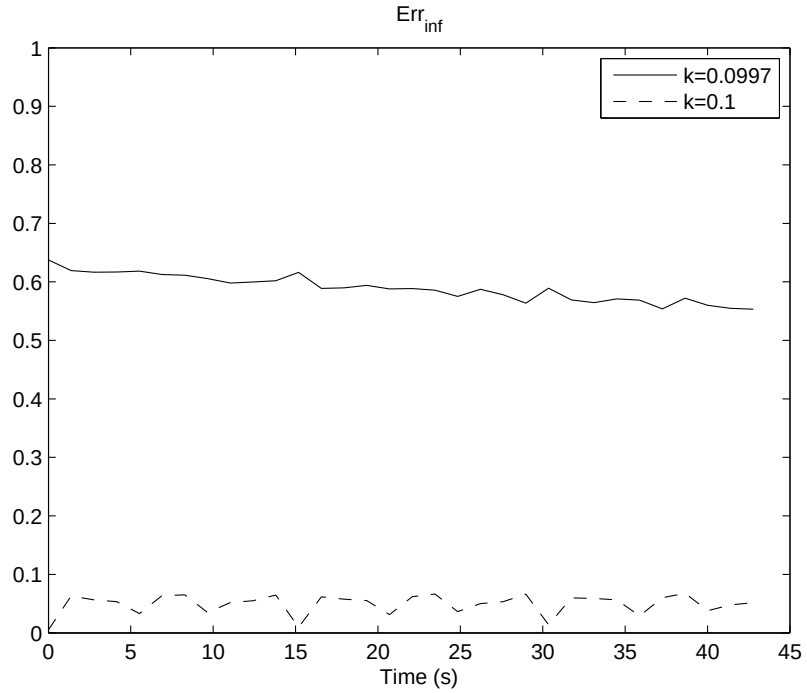

Figure 4.11: Err_∞ of the wave propagation using the actual $k = 0.1$ and the approximate $k = 0.0997$

4.3.4 Effect of non-optimal wavenumber

Since the wave number that we obtained had such a dramatic effect on the error functions we will investigate the consequence of different k values on the wave propagation.

Figure 4.12 shows the Err_2 functions as k varies at the initial ($t = 0$ seconds) and terminal time ($t = 42.78$ seconds). The error almost reaches 0 when $k = 0.1$, increases very steeply with only a small difference in k, and then starts to behave quite erratically when we are away from the actual k_0. With this plot we can clearly see the importance that k has in recreating the surface accurately, especially when the surface is going to be propagated.

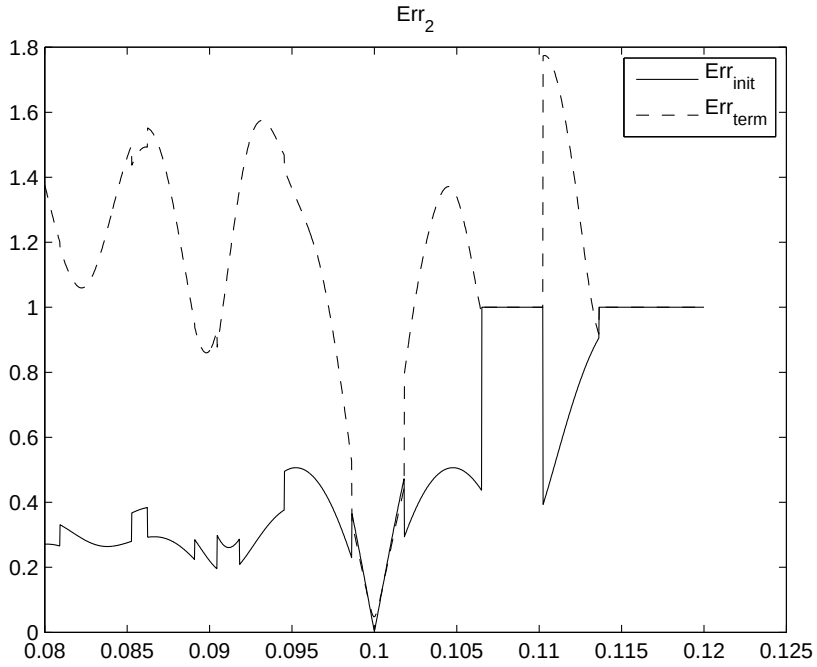

Figure 4.12: The initial and terminal L_2-norm as k varies

Chapter 5

Wave forecasting of an ocean surface

In this chapter we will attempt to use our method to forecast the propagation of an actual ocean surface.

5.1 The ocean surface

I have had access to three sets of ocean surfaces. A single surface is represented by a 128-by-128 point matrix where the distance between adjacent elements is 7.5 metres, each set then consists of 32 such matrices with 1.38 seconds separating each image. The data sets were obtained by OceanWaves by means of processing X-band radar data on a vessel offshore Norway.

As we can see in figure 5.1, the surface is considerably more complicated than what we've investigated so far. There is a multitude of different waves present in the image, all of whom have different wave vectors and, though not appearant from an overhead view, considerable difference in amplitude.

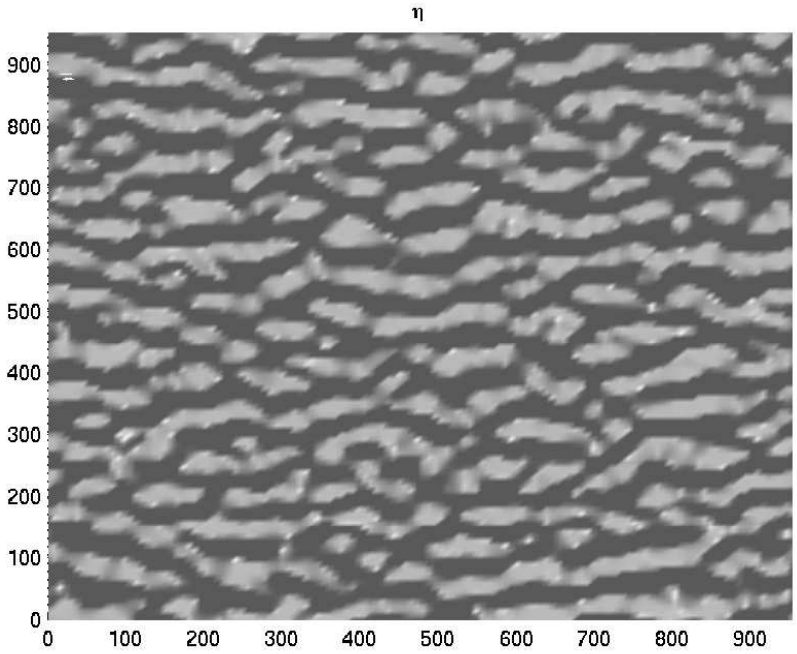

Figure 5.1: Observed ocean surface at $t = 0$

5.2 The Fourier transform

We will, yet again, use *Matlab*'s *fft* function to get the Fourier transform of the surface. In figures 5.2 and 5.3 we can see that the transform is quite a bit more chaotic than what we've previously seen.

The circular distribution tells us that there are waves moving along every direction. There is not a single distinguishable peak but rather several distinct high peaks in the image, this could imply that there are different waves, with different wave numbers, that are all influential on the final wave field. There is also a substantial part of the transform image that has small, yet non-zero, values, this shows that, in a small part of the spectrum, just about every wave number occurs, though alot of them occur quite rarely.

We have already seen, in section 4.3.3, that even a small error in the wave number could lead to dramatic differences in the error functions, this does not bode well for an actual surface for which there is no correct wave number.

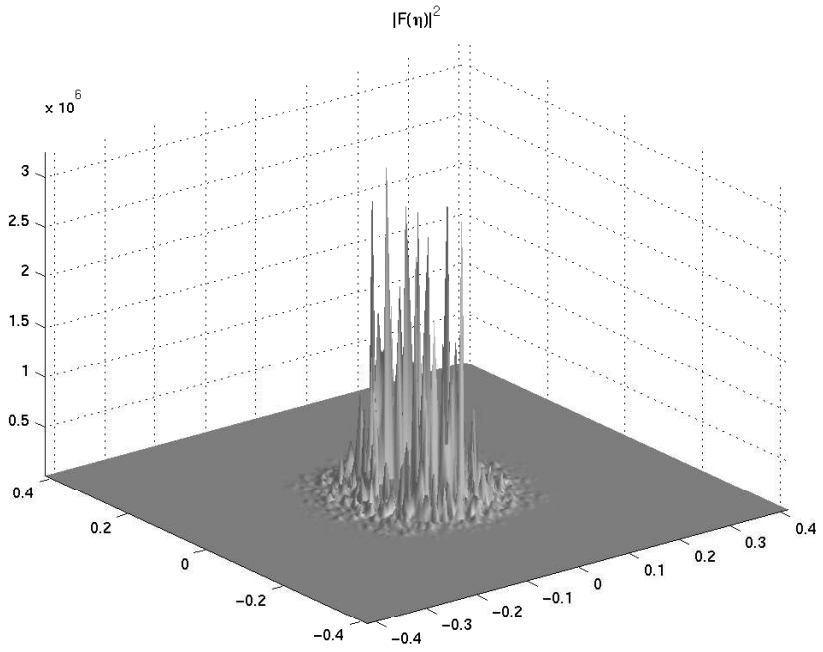

Figure 5.2: Fourier transform of ocean surface at $t = 0$ (isometric view)

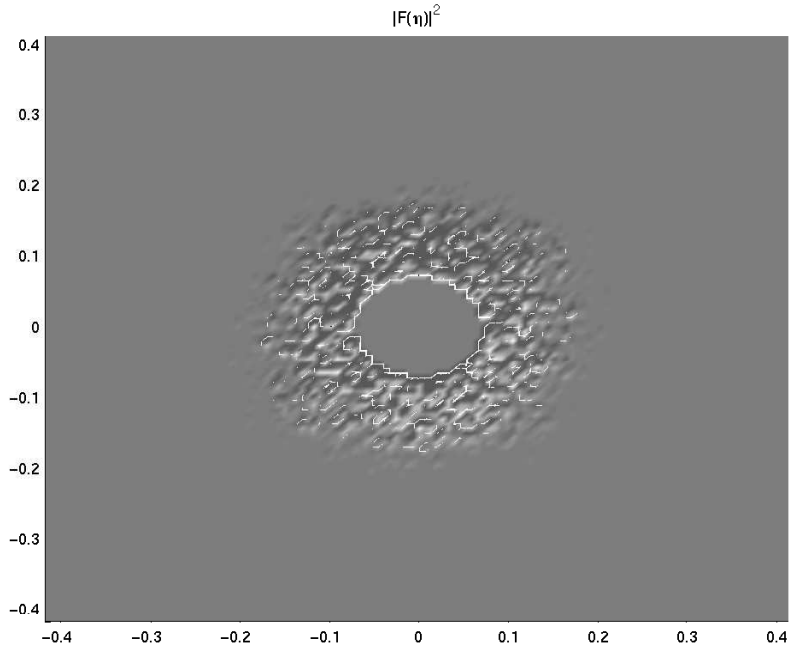

Figure 5.3: Fourier transform of ocean surface at $t = 0$ (top-down view)

5.2.1 The characteristic wave vector

To find the characteristic wave vector we will once again begin by calculating the angular and radial integrations of the Fourier transform so that we can find the characteristic wave number and the angle separately.

In figure 5.4 we can see that the angular integration has the same general form as the Fourier transform of the Draupner wave and in fact this form is present in the Fourier transform of almost every ocean surface. Calculating the characteristic wave number by the expected value method with exponent 2 then gives us that $k_c = 0.0978$.

The form of the radial integration in figure 5.5 shows that, in addition to what might be considered the 'main' characteristic angle at $\theta = 0$, there are also three peaks at around $\theta = 0.5$. Though the three peaks are only about half the height of the main peak, the combined area of them are quite influential and we get an angle of $\theta = 0.1129$ that lies between the two groups.

These values then gives us that $\mathbf{k}_c = (0.0971, 0.0110)$, $\omega = 0.9793$, $\mathbf{c} = (9.9531, 1.1281)$ and $\mathbf{C}_g = (4.9766, 0.5641)$.

5.2.2 The first order, first harmonic amplitude

As we saw in figure 5.2 and 5.3, the Fourier transform of an actual surface is very complicated and we will need some method to extract $\hat{B}$. We will, once again, use a low pass filter on $\hat{\eta}$ but we will need some criteria to decide the boundary. The criteria that I have used is to begin with all the $\mathbf{k}$ values that satisfy the inequality:

$$\frac{1}{5}\left|\hat{\eta}\right|^2\Big|_{\mathbf{k}=\mathbf{k}_c} \leq \left|\hat{\eta}\right|^2$$

I have then defined the smallest square area possible that is centred on $\mathbf{k_c}$ and encompass all the $\mathbf{k}$ values that satisfy the criteria. Setting all values outside of this area equal to zero and moving it to origo then leaves us with $\hat{B}$, as seen in figure 5.6, and by the inverse Fourier transform we get B, in figure 5.7. Using this method I manage to define an area that contains all of the highest and most influential peaks, while minimizing the amount of low magnitude 'noice'.

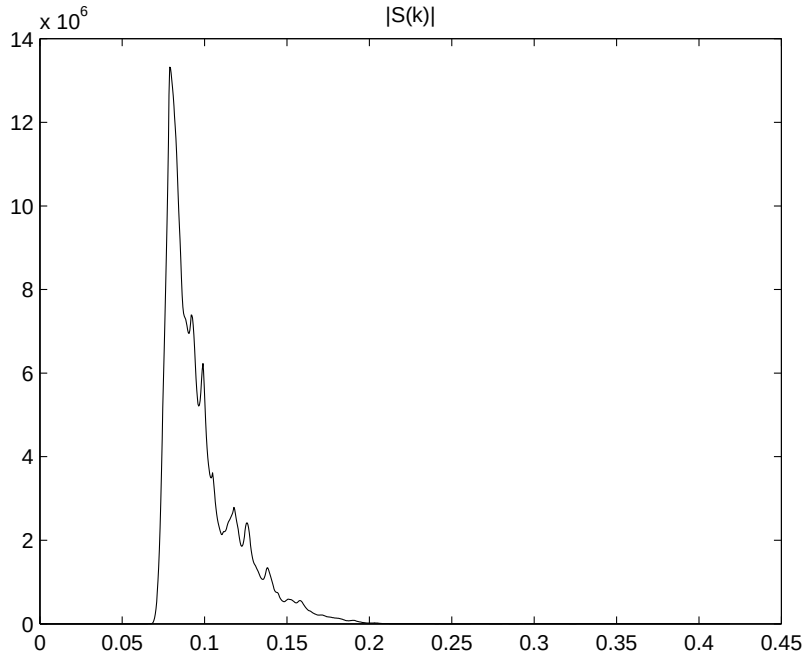

Figure 5.4: Angular integration of fourier transform

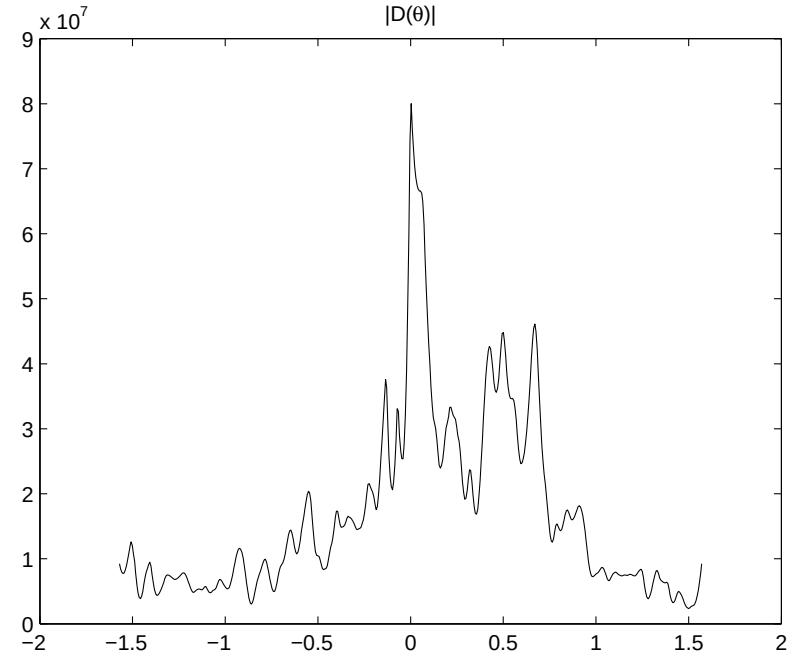

Figure 5.5: Radial integration of fourier transform

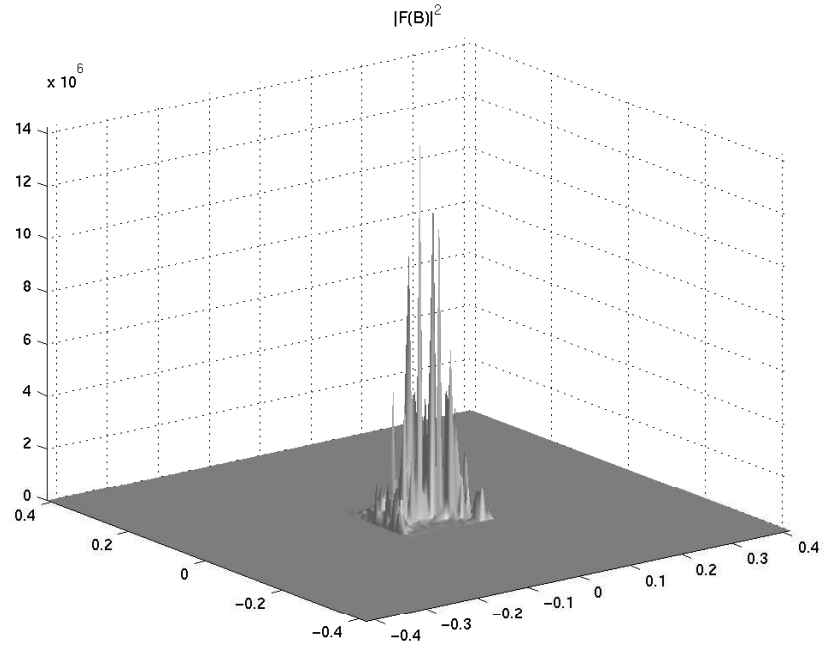

Figure 5.6: Fourier transform of first order amplitude at $t = 0$

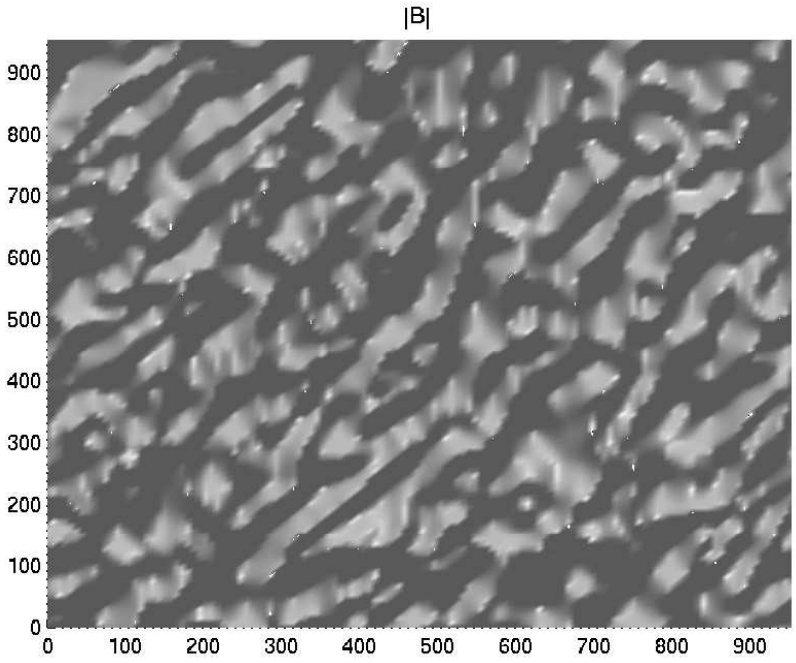

Figure 5.7: First order amplitude at $t = 0$

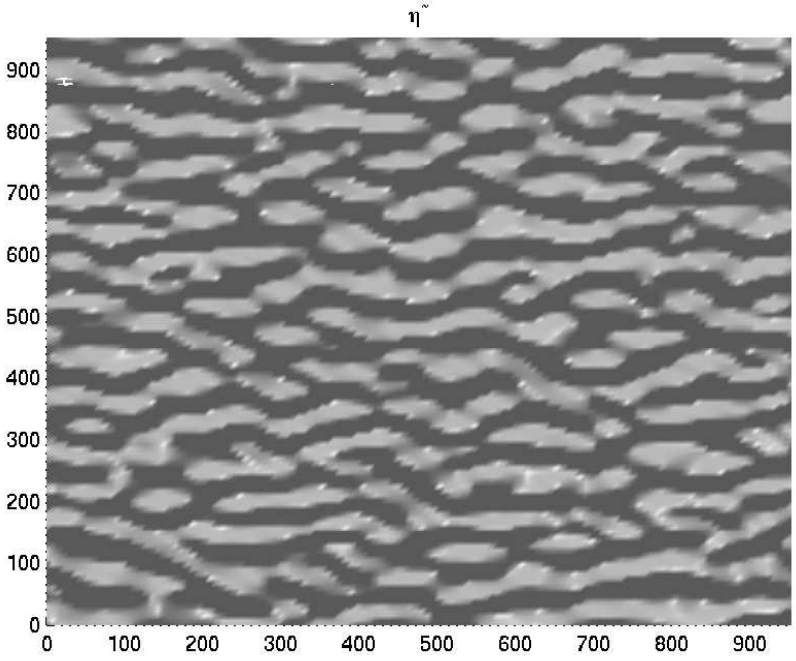

Figure 5.8: Recreated ocean surface at $t = 0$

5.3 Propagating the ocean surface

Using the reconstruction formulas from section 1.4 we can then obtain the higher order amplitudes and get the reconstructed surface in figure 5.8. As we can see the recreation is quite good, it is less detailed and a bit more long-crested, but the larger waves are all represented. The amplitude of the waves however are not always accurate and this is the main reason that the initial errors are $Err_1 = 0.4194$, $Err_2 = 0.4285$ and $Err_\infty = 0.7008$.

5.3.1 Error plots

The problems really start when we begin propagating the surface. As we can see in figure 5.9, in less than 10 seconds, all the errors are permanently above 1 and though they seem to halt around 1.3, this is an unacceptably high error.

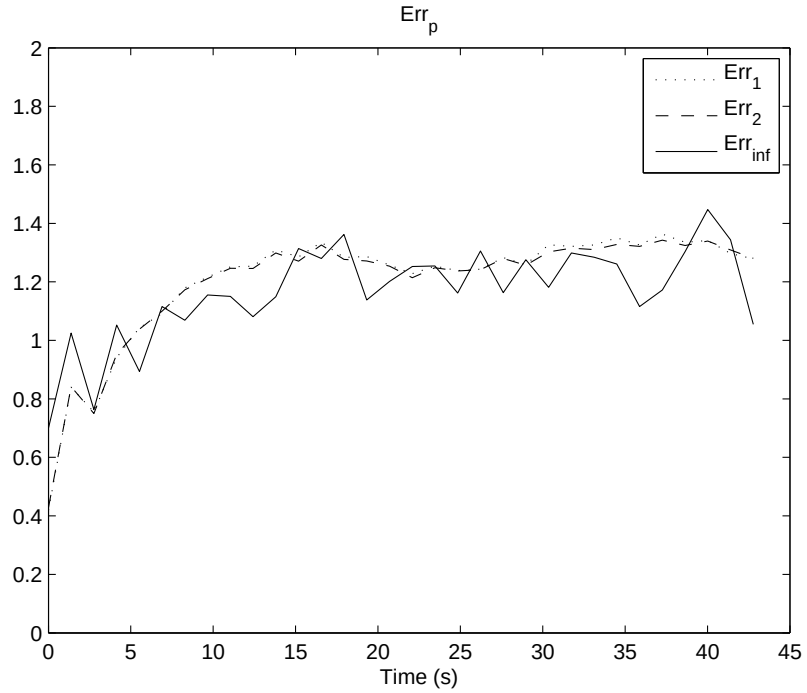

Figure 5.9: The Err_1, Err_2 and Err_∞ errors of the surface

5.3.2 Propagation using different wave numbers

Because of the critical importance of the wave number, and the high insecurity around the methods of obtaining it, we will investigate how well we can propagate the surface using different values of k_c.

Since the different errors have proven to be so similar, I will only include figure 5.10 showing the initial and terminal Err_2 plots, taken at $t = 0$ and $t = 42.78$ seconds, respectively. Unfortunelately, the plot of the terminal error shows that the propagation does not give an accurate result no matter which wave number we use.

But we can also see that the initial error has it's lowest value between $k_c \approx 0.07$ and $k_c \approx 0.095$ and that within that interval, the error is almost constant. If this is indicative of all ocean surfaces it is splendid news for our forecasting procedure. As we saw in figure 4.12, for the synthetic surface there was a steep increase in the error as we used a less than optimal k_c, with this surface however we have a large interval where every value is basically optimal. This would make obtaining the characteristic wave number less hazardous as the error would not be so unforgiving because of a small difference.

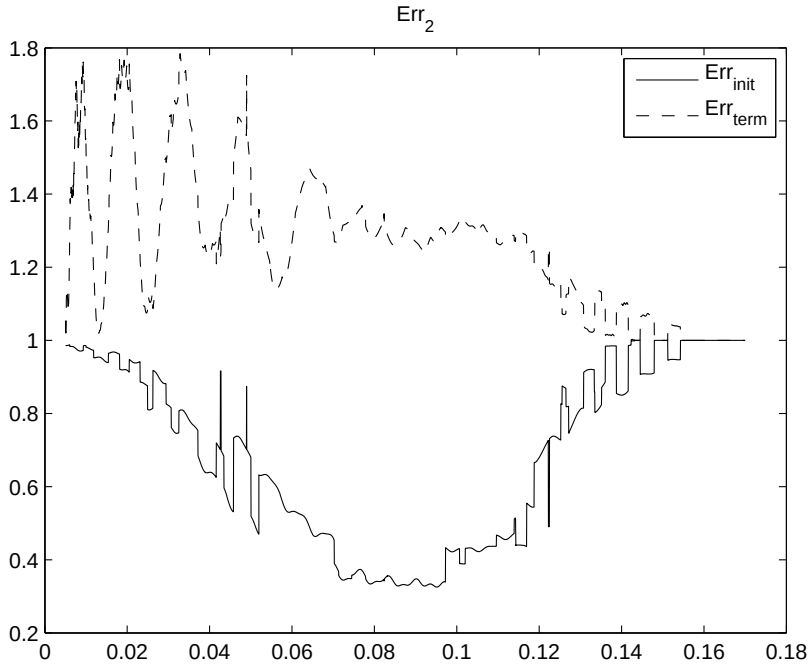

Figure 5.10: The initial and terminal Err_2 error as k varies

Since we saw in chapter 4 that the error was extremely steep around the optimal k_c, I have also done the same plot using a much finer grid to make sure that there aren't any hidden 'dips'. And as we can see in figure 5.11, the plot remains nearly constant throughout the interval.

5.3.3 Obtaining the wave number revisited

Now that we know where the optimal wave numbers lie, we can see that our value $k_c = 0.0978$ is actually slightly too high. This is a good opportunity to reexamine our different methods of obtaining the wave number.

I have once again used the expectation and integration method of obtaining the characteristic wave number using different exponents. In figure 5.12 we can see that the two methods still converge to a common value, which in this case is $k_c = 0.0785$.

We can see that in this instance, choosing a higher exponent would actually have been preferable and $n \approx 5$ would have landed us right in the middle of the optimal interval when using the expectation method.

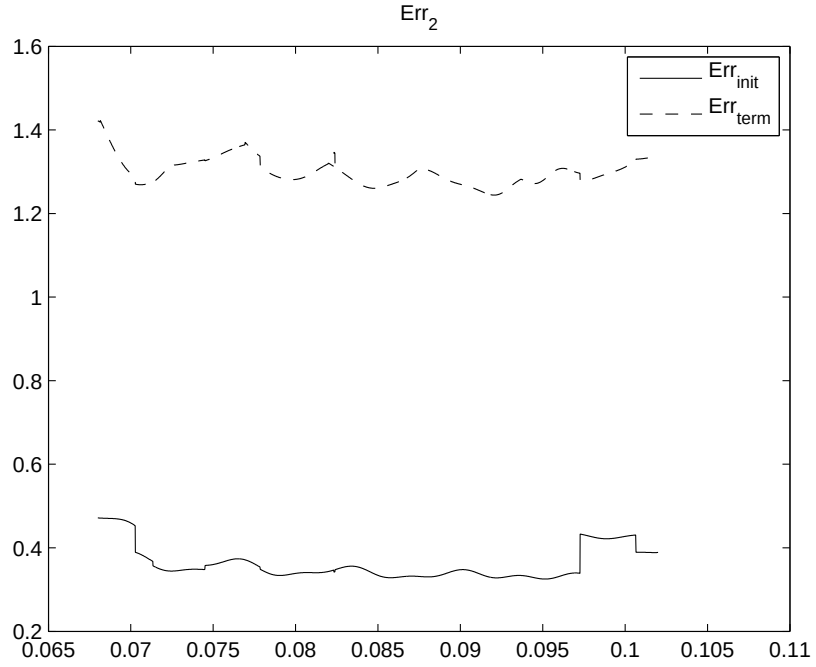

Figure 5.11: The Err_2 error as k varies, with a significantly finer grid

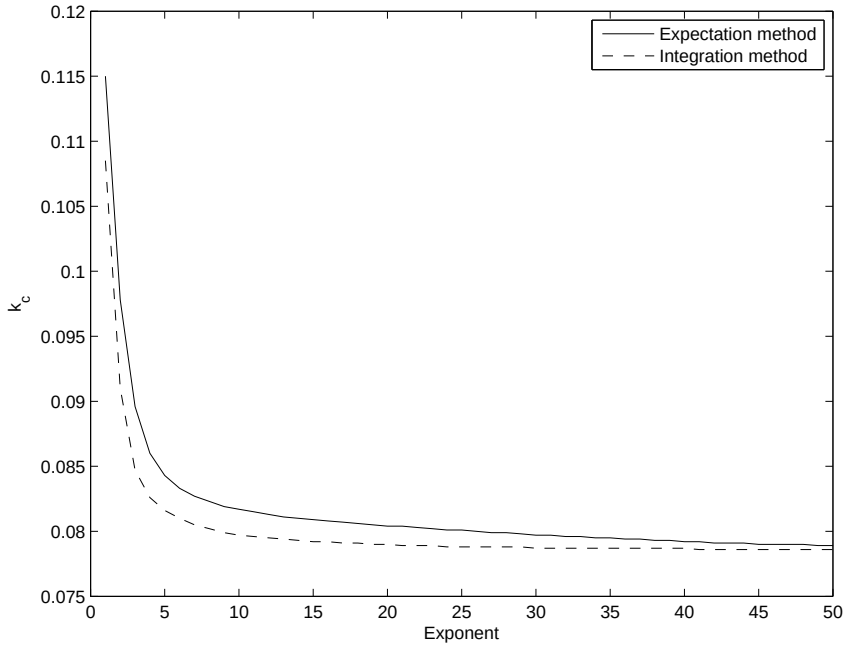

Figure 5.12: The expectation and integration methods, used with different exponents

Chapter 6

Final conclusions

Unfortunelately the method I have developed in this thesis has not proven itself to be accurate enough for predicting an actual ocean surface. The biggest problem is, as we saw already in chapter 4, to restore the amplitude B.

When working with the synthetic surface a relative error of only 0.003 in the estimate for the wave number turned out to have a major impact when recovering B and the error was magnified when propagating the surface. For an actual surface where there is no 'actual' wave number we will need a method that is not as unstable when it comes to different characteristic wave numbers.

For an actual forecasting system, we need to remember that not all the waves in a single surface image are necessarily moving in the same direction. For example, in figure 5.5 on page 53; the wave group with angle $\theta = 0$ and the one with angle $\theta = 0.5$ will quickly separate and for a destination sufficiently far away from the origin, we need only concern ourselves with the propagation of one of these groups.

One also needs to remember that from image to image, the wave number and hence the speed may vary. A working forecasting system will therefore need to continually update the wave field at the destination to make accurate predictions.

But before any of these methods are implemented, the most crucial thing is to improve or replace the method used for extracting the amplitude B. Without a reliable method for obtaining B, there is no hope of creating an accurate forecasting procedure.

Appendix A

Matlab scripts

A.1 Draupner wave

This is the script used in chapter 3. It reads the time series from file, performs a one-dimentional fourier transform and uses it to estimate B and ω_c. It also generates some figures including the original surface, the fourier transformed surface, the complex amplitudes of all orders and finally the reconstructed surface.

```
1  function [eta time B eta_bar B2 B3 omega k]...
2    =DraupnerWave(varargin)
3  if nargin==0
4    filename='freak.data';
5  else
6    filename=varargin{1};
7  end
8  X=load(filename)';        %load time-series from file
9  eta=mean(X)-X;            %convert coordinate system
10 T=20*60; H=70; g=9.81; %total time(s) and gravity
11 N=length(eta); Fs=T/N; %data points and point frequency
12 tstep=T/(N-1);            %time per data point
13 time=0:tstep:T;           %time interval
14 spectrum=(-N/2:N/2-1)/N*2*pi/Fs; %frequency spectrum
15
16 %surface plot
17 figure;
18   plot(time,eta,'k');
19   title(texlabel('eta(t)'))
20   xlabel('Time (s)')
```

```
21    ylabel('Surface elevation (m)')
22
23  %fourier transform of surface
24  eta_hat=2*fft(eta); N=ceil(length(eta_hat)/2);
25
26  %zoomed in view of the fourier transform
27  figure;
28    plot(spectrum(1280+(1:500)),...
29      abs(eta_hat(1:500)).^2,'k')
30    title(texlabel('2|F(eta)(omega)|^2'))
31    xlabel('Angular frequency (Hz)')
32    ylabel(texlabel('Frequency spectrum (m^{2} s)'))
33
34  temp=abs(eta_hat(1:N)).^2;
35  omega=sum(spectrum(1280+(1:N)).*temp)/sum(temp);
36  k=fzero(@(k)omega*omega-g*k*tanh(k*H),.05);%=0.0368
37
38  temp=2*fft(eta.*exp(-i*omega*time));
39  R=115; %radius
40  B_hat=zeros(size(temp));
41  B_hat(1:R)=temp(1:R);
42  B_hat(2560-R+(1:R))=temp(2560-R+(1:R));
43  B_hat=fftshift(B_hat);
44
45  int=1280+(-250:249);
46  figure;
47    plot(spectrum(int),abs(B_hat(int)).^2,'k')
48    title(texlabel('|F(B)(omega)|^2'))
49    xlabel('Angular frequency (Hz)')
50    ylabel(texlabel('Frequency spectrum (m^{2} s)'))
51
52  eps=k*sqrt(2*mean(eta.^2));%wave steepness
53  B=ifft(fftshift(B_hat));
54  eta_bar=zeros(size(B));
55  B20=k/2*B.^2;
56  B21=i*k/omega*B.*[B(2)-B(1) (B(3:end)-B(1:end-2))/2 ...
57    B(end)-B(end-1)]/tstep;
58  B2=B20+eps*B21;
59  B3=3*k/8*k^2*B.^3;
60
61  %surface with the wave envelope
62  figure;
63    plot(time,eta,'k');
64    hold on;
```

```
65    plot(time,abs(B),'k',time,-abs(B),'k','LineWidth',2);
66    hold off;
67    title(texlabel('eta+-|B|'));
68    xlabel('Time (s)')
69    ylabel('Surface elevation (m)')
70  %first harmonic amplitude
71  figure;
72    plot(time,abs(B),'k');
73    title(texlabel('|B(t)|'))
74    xlabel('Time (s)')
75    ylabel('Surface elevation (m)')
76  %set down
77  figure;
78    plot(time,abs(eta_bar),'k');
79    title(texlabel('|B_0(t)|'))
80    xlabel('Time (s)')
81    ylabel('Surface elevation (m)')
82  %second harmonic amplitude
83  figure;
84    plot(time,abs(B2),'k');
85    title(texlabel('|B_2(t)|'))
86    xlabel('Time (s)')
87    ylabel('Surface elevation (m)')
88  %third harmonic amplitude
89  figure;
90    plot(time,abs(B3),'k');
91    title(texlabel('|B_3(t)|'))
92    xlabel('Time (s)')
93    ylabel('Surface elevation (m)')
94
95  linSurf=real(B.*exp(i*(-time*omega)));
96
97  nLinSurf=eps*eta_bar+real(...
98     +       B .*exp(   i*(-time*omega))...
99     +eps*   B2.*exp(2*i*(-time*omega))...
100    +eps^2*B3.*exp(3*i*(-time*omega)));
101
102 err2_linear=...
103   sqrt(sum((linSurf-eta).^2))/sqrt(sum(eta.^2));
104 err2_nonlinear=...
105   sqrt(sum((nLinSurf-eta).^2))/sqrt(sum(eta.^2));
106
107 disp(['Err2: linear:   ' num2str(err2_linear)])
108 disp(['       nonlinear: ' num2str(err2_nonlinear)])
```

```
109
110 eta_tilde=nLinSurf;
111
112 figure;
113   plot(time,eta_tilde,'k');
114   title(texlabel('eta^~'));
115
116 figure;
117   plot(time,eta_tilde-eta,'k');
118   title(texlabel('eta^~-eta'));
```

A.2 Statoildata

This script is used to generate the results in chapters 4 and 5. It starts by either generating a synthetic surface (chapter 4) or reading actual ocean surface data from file (chapter 5). It then performs the operations necessary to recreate the surface, propagates it forwards in time and compares it with the actual surface data by calculating the L_1, L_2 and L_∞ norms of the difference.

```
1  function varargout=WavePropagation(varargin)
2  %
3
4  %%%%%%%%%%%%%%%%%%%%
5  %%% Check input %%%
6  %%%%%%%%%%%%%%%%%%%%
7  if nargin==0
8    %file='../Statoildata/20061010150748ifo.CI2';
9    %file='../Statoildata/20061010153433ifo.CI2';
10   file='../Statoildata/20061010155557ifo.CI2';
11 else
12   if isdir(varargin{1});
13     [filename filepath I]=uigetfile(fullfile(...
14       varargin{1},'*.CI2'),'Select surface data');
15     if ~I
16       error('User cancelled file select');
17     end
18     file=fullfile(filepath,filename);
19   else
20     file=varargin{1};
21   end
22 end
23
24
25 %%%%%%%%%%%%%%%%%%%%
26 %%% Read file %%%
27 %%%%%%%%%%%%%%%%%%%%
28 inFile=fopen(file);
29
30 [NXY NT DX DY RPT]=readHeader(inFile);
31 eta=readData(inFile,NXY,NT);
32
33 fclose(inFile);
34
35 %[NXY NT DX DY RPT eta]=fakesurface();
```

```
36
37 %%%%%%%%%%%%%%%%%%%%%%%%%%
38 %%% Quality assurance %%%
39 %%%%%%%%%%%%%%%%%%%%%%%%%%
40 %check if params have been initialized
41 if DX==DY
42   DXY=DX;
43 else
44   error('Area is not square');
45 end
46 origo = [1 1]+floor(NXY/2);
47 %%%%%%%%%%%%%%%%%%%%%%
48 %%% Process data %%%
49 %%%%%%%%%%%%%%%%%%%%%%
50 [k th]=computeK(eta{1},NXY,DXY,origo);
51 kx=k*cos(th);   ky=k*sin(th);
52 B1=cell(length(eta),1);
53
54 for n=1:NT
55   B1{n}=computeB1(eta,n,kx,ky,origo,NXY,DXY);
56 end
57
58 [L1 L2 Linf eta2]=...
59   timeShift(eta,B1{1},k,th,RPT,DXY,NXY,NT);
60
61 [K L1_start L1_end L2_start L2_end Linf_start...
62   Linf_end]=varyK(eta,k,th,RPT,origo,NXY,DXY,NT);
63
64 %%%%%%%%%%%%%%%%%%%%%%%%%%
65 %%% Create graphics %%%
66 %%%%%%%%%%%%%%%%%%%%%%%%%%
67
68 figs=logical([
69         1 %eta
70         1 %abs(B)
71         1 %abs(F(eta))^2
72         1 %abs(F(B))^2
73         1 %eta2
74         1 %Err_1
75         1 %Err_2
76         1 %Err_inf
77           %varyK
78         0 %Err_1
79         0 %Err_2
```

```
80          0 %Err_inf
81          ]);
82  if any(strcmp(who,'K'))
83    figs(9:11)=true;
84  end
85
86  xInt=DX*(0:(NXY-1)); yInt=DY*(0:(NXY-1));
87  if figs(1)
88    figure;
89      surfl(xInt,yInt,eta{1});
90      colormap('gray');  view(2);
91      shading interp;    axis tight
92      title(texlabel('eta'));
93  end
94
95  if figs(2)
96    figure;
97      surfl(xInt,yInt,abs(B1{1}));
98      colormap('gray');  view(2);
99      shading interp;    axis tight
100     title(texlabel('|B|'));
101 end
102
103 spectrum=(-NXY/2:NXY/2-1)/NXY*2*pi/DXY;
104 if figs(3)
105   figure;
106     surfl(spectrum,spectrum,...
107       abs(2*fftshift(fft2(eta{1}))).^2);
108     colormap('gray');  view(2);
109     shading interp;    axis tight
110     title(texlabel('2|F(eta)|^2'))
111 end
112
113 if figs(4)
114   figure;
115     surfl(spectrum,spectrum,...
116       abs(fftshift(fft2(B1{1}))).^2)
117     colormap('gray');  view(2);
118     shading interp;    axis tight
119     title(texlabel('|F(B)|^2'))
120 end
121
122 if figs(5)
123   figure;
```

```
124       surfl(xInt,yInt,eta2{1});
125       colormap('gray');  view(2);
126       shading interp;    axis tight
127       title(texlabel('eta~'))
128   end
129
130   T=(0:(NT-1))*RPT;
131   if figs(6)
132     figure;
133       plot(T,L1,'k');
134       title(texlabel('Err_1'));
135   end
136
137   if figs(7)
138     figure;
139       plot(T,L2,'k');
140       title(texlabel('Err_2'));
141   end
142
143   if figs(8)
144     figure;
145       plot(T,Linf,'k');
146       title(texlabel('Err_{inf}'));
147   end
148
149   %   varyK output
150   if figs(9)
151     figure;
152       hold on
153       plot(K,L1_start,'k',K,L1_end,'k--');
154       title(texlabel('Err_1'));
155       legend(texlabel('Err_{init}'),...
156         texlabel('Err_{term}'))
157       hold off
158   end
159
160   if figs(10)
161     figure;
162       hold on
163       plot(K,L2_start,'k',K,L2_end,'k--');
164       title(texlabel('Err_2'));
165       legend(texlabel('Err_{init}'),...
166         texlabel('Err_{term}'))
167       hold off
```

```
168 end
169
170 if figs(11)
171   figure;
172     hold on
173     plot(K,Linf_start,'k',K,Linf_end,'k--');
174     title(texlabel('Err_{inf}'));
175     legend(texlabel('Err_{init}'),...
176       texlabel('Err_{term}'))
177     hold off
178 end
179
180 %%%%%%%%%%%%%%%%%%%
181 %%% Set output %%%
182 %%%%%%%%%%%%%%%%%%%
183 varargout{1}=eta;
184 varargout{2}=B1;
185 varargout{3}=k;
186 varargout{4}=th;
187 varargout{5}=L1;
188 varargout{6}=L2;
189 varargout{7}=Linf;
190
191 %=========================================================
192 function [NXY NT DX DY RPT eta]=fakesurface()
193 DX=7.5;DY=7.5;NXY=128; NT=32; RPT=1.38;
194
195 [y x]=meshgrid(DX*(0:(NXY-1)));
196
197 k=0.1; th=acos(.8);
198 kx=k*cos(th); ky =k*sin(th);
199 omega=sqrt(k*9.81);
200 t1=0;
201
202 B_t1=ones(NXY,NXY);
203 eta=cell(NT,1);
204
205 eta{1}=real(B_t1.*exp(i*(kx*x+ky*y-t1*omega)));
206 eps=k*sqrt(2*mean(mean(eta{1}.^2)));
207
208 B=evolveB(B_t1,omega,kx,ky,eps,NT,NXY,DX,RPT);
209
210 for n=2:NT
211   t=t1+RPT*(n-1);
```

```
212     eta{n}=real(B{n}.*exp(i*(kx*x+ky*y-t*omega)));
213 end
214
215 %=========================================================
216 function [k th]=computeK(eta,NXY,DXY,origo)
217 %compute surface in polar coords
218
219 Feta=fftshift(fft2(eta));
220
221 K_length=1000; Th_length=500;
222
223 % Range for k and theta values
224 th=linspace(-pi/2,pi/2,Th_length);
225 k=linspace(0,64*2*pi/(NXY*DXY),K_length)';
226
227 % new meshgrid values
228 Th=repmat(th,K_length,1);
229 K=repmat(k,1,Th_length);
230
231 % old range for kx and ky
232 Kx=((1:NXY)-origo(1))*2*pi/(NXY*DXY);     Ky=Kx;
233
234 % new meshgrid in cartesian
235 [Ky1 Kx1]=pol2cart(Th,K);
236
237 % F(eta) in polar coords
238 % (set to 0 outside original range)
239 Feta_pol= interp2(Kx,Ky,Feta,Kx1,Ky1,'linear',0);
240
241 n=2;
242 %integrate over theta...
243 S=sum(abs(Feta_pol).^n,2)-...
244     abs(Feta_pol(:,1)).^n/2-abs(Feta_pol(:,end)).^n/2;
245 S=k.*S; %Jacobian determinant
246
247 %figure; plot(k,S,'k'); title(texlabel('|S(k)|'))
248
249 %...then over k
250 D=sum(abs(Feta_pol).^n,1)-...
251     abs(Feta_pol(1,:)).^n/2-abs(Feta_pol(end,:)).^n/2;
252
253 %figure; plot(th,D,'k'); title(texlabel('|D(theta)|'))
254
255 k=findCentre(S,k);
```

```
256
257 th=findCentre(D,th);
258
259 k=0.085
260
261 %=========================================================
262 function x0=findCentre2(f,X)
263 % finds the value x0 that fulfills the condition:
264 %
265 % /x0            /infinity
266 % |              |
267 % | f(x) dx  =   | f(x) dx
268 % |              |
269 %/ -infinity   / x0
270
271 f=f(:)';
272 N=length(f);
273 %trapezoidal integration for [1,n], n=1:N
274 F=[0 cumsum(f(1:(N-1))+f(2:N))/2];
275
276 %integer part
277 n=find(F<F(N)/2,1,'last');
278 %decimal part by linear interpolation
279 m=(F(N)/2-F(n))/(F(n+1)-F(n));
280
281 x0=X(n)*(1-m)+X(n+1)*m;
282
283 %=========================================================
284 function x0=findCentre(f,X)
285 %computes E(f/sum(f))
286 f=f(:); X=X(:);
287
288 x0=sum(X.*f)/sum(f);
289
290 %=========================================================
291 function [eta]=readData(inFile,NXY,NT)
292
293 eta=cell(NT,1);
294 for n=1:NT
295   eta{n}=...
296     reshape(fread(inFile,NXY*NXY,'float32'),NXY,NXY)';
297 end
298 if ~isempty(fread(inFile))
299   error('End of file was NOT reached!!')
```

```
300 end
301
302 %=========================================================
303 function [NXY NT DX DY RPT]=readHeader(inFile)
304
305 %initializing variables in case of read error
306 NXY=-1;NT=-1;DX=-1;DY=-1;RPT=-1;
307
308 EOH=0;
309 while ~EOH
310   line = fgetl(inFile);
311   [var count errmsg nIdx]=sscanf(line,'%s',1);
312
313   if any(strcmp(var,{'NXY','NT','DX','DY','RPT'}))
314     val =sscanf(line(nIdx:end),'%f',1);
315     eval([var '=' num2str(val) ';']);
316   elseif strcmp(var,'EOH')
317     EOH=1;
318   end
319 end
320
321 %=========================================================
322 function B=computeB1(eta,n,kx,ky,origo,NXY,DXY)
323 [y x]=meshgrid(DXY*(0:(NXY-1)));
324
325 temp=2*fftshift(fft2(eta{n}.*exp(-i*(kx*x+ky*y))));
326
327 % k0 shifted to 0
328 eta_hat=temp;xIdx=origo(1);yIdx=origo(2);
329
330
331 %Chooses a rectangle centred on k0 where all values
332 %on the outside are less than M/5
333 M=max(max(abs(eta_hat)));
334 [I J]=find(abs(eta_hat(origo(1):end,:))>M/5);
335 I=I+origo(1)-1;
336
337 I=max(max(I)-xIdx,xIdx-min(I));
338 J=max(max(J)-yIdx,yIdx-min(J));
339 I=max(I,J); J=I;
340
341
342 B_hat=zeros(size(eta_hat));
343 I=-I:I;  J=-J:J;
```

```
344
345 B_hat(origo(1)+I,origo(2)+J)=eta_hat(xIdx+I,yIdx+J);
346
347 B=ifft2(fftshift(B_hat));
348
349 %=========================================================
350 function B=evolveB(B_0,omega,kcx,kcy,eps,NT,NXY,DXY,dt)
351 B=cell(NT,1); B{1}=B_0;
352 fB_0=fftshift(fft2(B_0));
353
354 spectrum=(-NXY/2:NXY/2-1)/NXY*2*pi/DXY;
355 [ky kx]=meshgrid(spectrum);
356 Kx=kx-kcx;  Ky=ky-kcy; k=sqrt(kx.^2+ky.^2);
357
358 eps1=eps;
359 eps2=eps^2;
360
361 h=omega./(2*k.^2).*((kx.*Kx+ky.*Ky)...
362   -eps1./(4*k.^2).*(...
363       (3*kx.^2-2*k.^2).*Kx.^2  ...
364     +(3*ky.^2-2*k.^2).*Ky.^2  ...
365     +6*kx.*ky.*Kx.*Ky)...
366   -eps2./(8*k.^4).*(...
367       (6*k.^2-7*kx.^2).*kx.*Kx.^3  ...
368     +(6*k.^2-7*ky.^2).*ky.*Ky.^3  ...
369     +(6*k.^2-21*ky.^2).*kx.*Kx.*Ky.^2  ...
370     +(6*k.^2-21*kx.^2).*ky.*Kx.^2.*Ky  ));
371
372 for n=2:NT
373   e=exp(-i*eps*h*dt*(n-1));
374   e(~isfinite(e))=1;%remove NaN at k=0
375   fB=fB_0.*e;
376   B{n}=ifft2(fftshift(fB));
377 end
378
379
380
381 %=========================================================
382 function [L1 L2 Linf eta2]...
383   =timeShift(eta,B,k,th,RPT,DXY,NXY,NT)
384
385 N=length(eta); eta2=cell(N,1);
386 L1=zeros(N,1); L2=zeros(N,1); Linf=zeros(N,1);
387 kx=k*cos(th); ky=k*sin(th);
```

```
388 omega=sqrt(k*9.81);
389 eps=k*sqrt(2*mean(mean(eta{1}.^2)));
390 Cg=omega/(2*k); t1=0;
391 [y x]=meshgrid(DXY*(0:NXY-1));
392 B=evolveB(B,omega,kx,ky,eps,NT,NXY,DXY,RPT);
393
394 for n=1:N
395   t=t1+(n-1)*RPT;
396
397   eta2{n}=makeSurface(B{n},eps,kx,ky,t,...
398       x+Cg*t*cos(th),y+Cg*t*sin(th),DXY);
399   [L1(n) L2(n) Linf(n)]=...
400     LpNorm(eta{n},eta2{n},t-t1,Cg,th,DXY,NXY);
401 end
402
403 %=========================================================
404 function [L1 L2 Linf]...
405   =LpNorm(eta,eta2,dt,Cg,th,DXY,NXY)
406
407 range=DXY*(0:NXY-1);
408 [y x]=meshgrid(range);
409 x2=range+Cg*dt*cos(th);
410 y2=range+Cg*dt*sin(th);
411
412 %interpolate from the B grid to the radar grid
413 eta2 = interp2(y2,x2,eta2,y,x);
414
415 %identify the area we have comparable data for
416 J=find(isfinite(eta2(end,:)));
417 I=find(isfinite(eta2(:,J(1))));
418
419 %only include the data that will be compared
420 eta  = eta(I,J);
421 eta2 = eta2(I,J);
422
423 %normalizing constants
424 norm1=sum(sum(abs(eta)));
425 norm2=sqrt(sum(sum(eta.^2)));
426 normInf=max(max(abs(eta)));
427
428 L1=sum(sum(abs(eta-eta2)));
429 L2=sqrt(sum(sum(abs(eta-eta2).^2)));
430 Linf=max(max(abs(eta-eta2)));
431
```

```
L1=L1/norm1; L2=L2/norm2; Linf=Linf/normInf;

%=========================================================
function [K L1_start L1_end L2_start L2_end...
   Linf_start Linf_end] =...
   varyK(eta,k,th,RPT,origo,NXY,DXY,NT)
N=1000;
%K=k+(-N:N)*k/(5*N);
K=linspace(0,2*k,4*N+1);


%removes k values leading to errors
K=K(K>0.005 & K<0.3);

[y x]=meshgrid(DXY*(0:NXY-1));

L1_start=zeros(length(K),1);
L1_end=zeros(length(K),1);
L2_start=zeros(length(K),1);
L2_end=zeros(length(K),1);
Linf_start=zeros(length(K),1);
Linf_end=zeros(length(K),1);

for n=1:length(K)

   k=K(n);
   omega=sqrt(k*9.81); t1=0;
   Cg=omega/(2*k);
   eps=k*sqrt(2*mean(mean(eta{1}.^2)));
   kx=k*cos(th);   ky=k*sin(th);
   B=computeB1(eta,1,kx,ky,origo,NXY,DXY);
   B=evolveB(B,omega,kx,ky,eps,NT,NXY,DXY,RPT);

   t=t1;
   eta2=makeSurface(B{1},eps,kx,ky,t,x,y,DXY);
   [L1_start(n) L2_start(n) Linf_start(n)]...
     =LpNorm(eta{1},eta2,0,Cg,th,DXY,NXY);

   t=t1+(NT-1)*RPT;
   eta2=makeSurface(B{NT},eps,kx,ky,t,...
        x+Cg*(t-t1)*cos(th),y+Cg*(t-t1)*sin(th),DXY);
   [L1_end(n) L2_end(n) Linf_end(n)]...
     =LpNorm(eta{end},eta2,t-t1,Cg,th,DXY,NXY);
end
```

```
476
477 %=========================================================
478 function df=diffx(f)
479
480 df = [f(2,:)-f(1,:); ...
481   (f(3:end,:)-f(1:end-2,:))/2; f(end,:)-f(end-1,:)];
482
483 %=========================================================
484 function df=diffy(f)
485
486 df = diffx(f')';
487
488 %=========================================================
489 function eta=makeSurface(B_t,eps,kx,ky,t,x,y,DXY)
490
491 k=sqrt(kx^2+ky^2);
492 omega=sqrt(k*9.81);
493
494 eta_bar=0;
495 B20=k/2*B_t.^2;
496 B21=-i/(2*k)*B_t.*(kx*diffx(B_t)+ky*diffy(B_t))/DXY;
497 B2=B20+eps*B21;
498 B3=3/8*k^2*B_t.^3;
499
500 eta=eps*  eta_bar+real(...
501     +       B_t.*exp(   i*(kx*x+ky*y-t*omega))...
502     +eps*   B2 .*exp(2*i*(kx*x+ky*y-t*omega))...
503     +eps^2*B3 .*exp(3*i*(kx*x+ky*y-t*omega)));
```

Appendix B

Maple script

For completeness I include the maple script used to generate and solve the Schrödinger equations. The equations are automatically generated and the programme was run iteratively, solving a higher order each time.

```
assume(0<eps,eps<infinity,H>0,0<k,k<infinity,0<omega,
  omega<infinity,0<s,s<infinity,0<kx,kx<infinity,0<ky,
  ky<infinity,0<x,x<infinity,0<y,y<infinity,0<z,
  z<infinity,0<t,t<infinity,0<x1,x1<infinity,0<y1,
  y1<infinity,0<t1,t1<infinity):

del:=(f)->Vector([diffx(f),diffy(f),diffz(f)]):
lap:=(f)->
  diffx(diffx(f))+diffy(diffy(f))+diffz(diffz(f)):
dot:=(U,V)->
  LinearAlgebra:-BilinearForm(U,V,conjugate=false):

subsdisp:=(f)->algsubs(omega^2 = k*g,f):
subsk:=(f)->algsubs(kx^2+ky^2=k^2,f,exact):
subsCg:=(f)->algsubs(diff(B(x1,y1,t1),t1)=
  -(diff(B(x1, y1, t1), x1)*kx
  +diff(B(x1, y1, t1), y1)*ky)*(omega/(2*k^2)),f):
subsCcg:=(f)->algsubs(diff(Bc(x1,y1,t1),t1)=
  -(diff(Bc(x1, y1, t1), x1)*kx
  +diff(Bc(x1, y1, t1), y1)*ky)*(omega/(2*k^2)),f):

#Governing equations
tempL:=lap(varphi(x,y,z,t,x1,y1,z1,t1)):
tempK:=difft(vareta(x,y,t,x1,y1,t1))
  +dot(del(varphi(x,y,z,t,x1,y1,z1,t1)),
```

```
26      del(vareta(x,y,t,x1,y1,t1)))
27    -diffz(varphi(x,y,z,t,x1,y1,z1,t1)):
28  tempD:=difft(varphi(x,y,z,t,x1,y1,z1,t1))
29    +g*vareta(x,y,t,x1,y1,t1)
30    +1/2*dot(del(varphi(x,y,z,t,x1,y1,z1,t1)),
31      del(varphi(x,y,z,t,x1,y1,z1,t1))):
32  tempB:=diffz(varphi(x,y,z,t,x1,y1,z1,t1)):
33
34  #Slow modulation variables
35  #difft:=(f)->diff(f,t):
36  #diffx:=(f)->diff(f,x):
37  #diffy:=(f)->diff(f,y):
38  #diffz:=(f)->diff(f,z):
39
40  difft:=(f)->diff(f,t)+eps*diff(f,t1):
41  diffx:=(f)->diff(f,x)+eps*diff(f,x1):
42  diffy:=(f)->diff(f,y)+eps*diff(f,y1):
43  diffz:=(f)->diff(f,z)+eps*diff(f,z1):
44
45  #Maclaurin expansion
46  tempK:=eval(eval(tempK,varphi(x,y,z,t,x1,y1,z1,t1)=
47      varphi(x,y,z,t,x1,y1,z1,t1)
48    +tempEta
49    *diffz(varphi(x,y,z,t,x1,y1,z1,t1))
50    +(1/2)*tempEta^2
51    *diffz(diffz(varphi(x,y,z,t,x1,y1,z1,t1)))),
52    tempEta=vareta(x,y,t,x1,y1,t1)):
53  tempD:=eval(eval(tempD,varphi(x,y,z,t,x1,y1,z1,t1)
54    =varphi(x,y,z,t,x1,y1,z1,t1)
55    +tempEta
56    *diffz(varphi(x,y,z,t,x1,y1,z1,t1))
57    +(1/2)*tempEta^2
58    *diffz(diffz(varphi(x,y,z,t,x1,y1,z1,t1)))),
59    tempEta=vareta(x,y,t,x1,y1,t1)):
60
61  #eta,phi~eps
62  tempL:=eval(tempL/eps,[
63    varphi(x,y,z,t,x1,y1,z1,t1)
64    =eps*varphi(x,y,z,t,x1,y1,z1,t1),
65    vareta(x,y,t,x1,y1,t1)
66    =eps*vareta(x,y,t,x1,y1,t1)]):
67  tempK:=eval(tempK/eps,[
68    varphi(x,y,z,t,x1,y1,z1,t1)
69    =eps*varphi(x,y,z,t,x1,y1,z1,t1),
```

```
70    vareta(x,y,t,x1,y1,t1)
71    =eps*vareta(x,y,t,x1,y1,t1)]):
72  tempD:=eval(tempD/eps,[
73    varphi(x,y,z,t,x1,y1,z1,t1)
74    =eps*varphi(x,y,z,t,x1,y1,z1,t1),
75    vareta(x,y,t,x1,y1,t1)
76    =eps*vareta(x,y,t,x1,y1,t1)]):
77  tempB:=eval(tempB/eps,[
78    varphi(x,y,z,t,x1,y1,z1,t1)
79    =eps*varphi(x,y,z,t,x1,y1,z1,t1),
80    vareta(x,y,t,x1,y1,t1)
81    =eps*vareta(x,y,t,x1,y1,t1)]):
82
83  #Form assumption
84  vareta:=(x,y,t,x1,y1,t1)  ->eps*vareta_bar(x1,y1,t1)
85    +1/2*(B(x1,y1,t1)*exp(I*theta(x,y,t))
86    +eps*B2(x1,y1,t1)*exp(2*I*theta(x,y,t))
87    +eps^2*B3(x1,y1,t1)*exp(3*I*theta(x,y,t))
88    +Bc(x1,y1,t1)*exp(-I*theta(x,y,t))
89    +eps*Bc2(x1,y1,t1)*exp(-2*I*theta(x,y,t))
90    +eps^2*Bc3(x1,y1,t1)*exp(-3*I*theta(x,y,t))):
91  varphi:=(x,y,z,t,x1,y1,z1,t1)->
92    eps*varphi_bar(z,x1,y1,z1,t1)
93    +1/2*(A(z,x1,y1,t1)*exp(I*theta(x,y,t))
94    +eps*A2(z,x1,y1,t1)*exp(2*I*theta(x,y,t))
95    +eps^2*A3(z,x1,y1,t1)*exp(3*I*theta(x,y,t))
96    +Ac(z,x1,y1,t1)*exp(-I*theta(x,y,t))
97    +eps*Ac2(z,x1,y1,t1)*exp(-2*I*theta(x,y,t))
98    +eps^2*Ac3(z,x1,y1,t1)*exp(-3*I*theta(x,y,t))):
99  theta:=(x,y,t)->kx*x+ky*y-omega*t:
100
101
102 #Perturbation
103 A :=(z,x1,y1,t1)->A10(z,x1,y1,t1)+eps*A11(z,x1,y1,t1)
104   +eps^2*A12(z,x1,y1,t1)+eps^3*A13(z,x1,y1,t1):
105 A2:=(z,x1,y1,t1)->A20(z,x1,y1,t1)+eps*A21(z,x1,y1,t1):
106 B2:=(x1,y1,t1)->B20(x1,y1,t1)+eps*B21(x1,y1,t1):
107 varphi_bar:=(z,x1,y1,z1,t1)->varphi_bar0(z,x1,y1,z1,t1)
108   +eps*varphi_bar1(z,x1,y1,z1,t1):
109 vareta_bar:=(x1,y1,t1)->vareta_bar0(x1,y1,t1)
110   +eps*vareta_bar1(x1,y1,t1):
111
112
113 Ac :=(z,x1,y1,t1)->Ac10(z,x1,y1,t1)
```

```
  +eps*Ac11(z,x1,y1,t1)+eps^2*Ac12(z,x1,y1,t1):
Ac2:=(z,x1,y1,t1)->Ac20(z,x1,y1,t1)
  +eps*Ac21(z,x1,y1,t1):
Bc2:=(x1,y1,t1)->Bc20(x1,y1,t1)+eps*Bc21(x1,y1,t1):

#Gather terms according to order
laplace:=0:
for n from 0 to 3 do
  laplace:= laplace+coeff(tempL, eps, 3-n)*eps^(3-n)
end do:
lap1:=coeff(laplace,eps,0):
lap2:=coeff(laplace,eps,1):
lap3:=coeff(laplace,eps,2):
lap4:=coeff(laplace,eps,3):

Kin:=0:
for n from 0 to 3 do
  Kin := Kin+coeff(tempK, eps, 3-n)*eps^(3-n)
end do:
Kin1:=coeff(Kin,eps,0):
Kin2:=coeff(Kin,eps,1):
Kin3:=coeff(Kin,eps,2):
Kin4:=coeff(Kin,eps,3):

Dyn:=0:
for n from 0 to 3 do
  Dyn := Dyn+coeff(tempD, eps, 3-n)*eps^(3-n)
end do:
Dyn1:=coeff(Dyn,eps,0):
Dyn2:=coeff(Dyn,eps,1):
Dyn3:=coeff(Dyn,eps,2):
Dyn4:=coeff(Dyn,eps,3):

Bot:=0:
for n from 0 to 3 do
  Bot := Bot+coeff(tempB, eps, 3-n)*eps^(3-n)
end do:
Bot1:=coeff(Bot,eps,0):
Bot2:=coeff(Bot,eps,1):
Bot3:=coeff(Bot,eps,2):
Bot4:=coeff(Bot,eps,3):

#O(eps)
A10:=(z,x1,y1,t1)->-I*B(x1,y1,t1)*omega*exp(k*z)/k:
```

```
158 Ac10:=(z,x1,y1,t1)->I*Bc(x1,y1,t1)*omega*exp(k*z)/k:
159
160 #Dispersion relation
161 #omega^2=g*k
162
163 #O(eps^2)
164
165 #lap2 & bot2
166 A11:=(z,x1,y1,t1)->A11_C(x1,y1,t1)*exp(k*z)
167   -(omega/k^2)*(kx*diff(B(x1,y1,t1),x1)
168   +ky*diff(B(x1,y1,t1),y1))*z*exp(k*z):
169 A20:=(z,x1,y1,t1)->A20_C(x1,y1,t1)*exp(2*k*z):
170 varphi_bar0:=(z,x1,y1,z1,t1)->varphi0_C(x1,y1,z1,t1):
171
172 Ac11:=(z,x1,y1,t1)->Ac11_C(x1,y1,t1)*exp(k*z)
173   -(omega/k^2)*(kx*diff(Bc(x1,y1,t1),x1)
174   +ky*diff(Bc(x1,y1,t1),y1))*z*exp(k*z):
175 Ac20:=(z,x1,y1,t1)->Ac20_C(x1,y1,t1)*exp(2*k*z):
176
177 #dyn2
178 #A11_C:=(x1,y1,t1)->-(diff(B(x1, y1, t1), t1))/k:
179 #A20_C:=(x1,y1,t1)->(1/4*I)*omega*(k*B(x1, y1, t1)^2
180   -2*B20(x1, y1, t1))/k:
181 vareta_bar0:=(x1,y1,t1)->0:
182
183 #Ac11_C:=(x1,y1,t1)->-(diff(Bc(x1, y1, t1), t1))/k:
184 #Ac20_C:=(x1,y1,t1)->-(1/4*I)*omega*(k*Bc(x1, y1, t1)^2
185   -2*Bc20(x1, y1, t1))/k:
186
187 #kin2
188 A11_C:=(x1,y1,t1)->(omega*kx*(diff(B(x1, y1, t1), x1))
189   +(diff(B(x1, y1, t1), t1))*k^2
190   +omega*ky*(diff(B(x1, y1, t1), y1)))/k^3:
191 #A20_C:=(x1,y1,t1)->(1/2*I)*omega*(k*B(x1, y1, t1)^2
192   -2*B20(x1, y1, t1))/k:
193
194 Ac11_C:=(x1,y1,t1)->(omega*kx*(diff(Bc(x1, y1, t1),x1))
195   +(diff(Bc(x1, y1, t1), t1))*k^2
196   +omega*ky*(diff(Bc(x1, y1, t1), y1)))/k^3:
197 #Ac20_C:=(x1,y1,t1)->-(1/2*I)*omega*(k*Bc(x1, y1, t1)^2
198   -2*Bc20(x1, y1, t1))/k:
199
200 #surf2
201 B20:=(x1, y1, t1)->(1/2)*k*B(x1, y1, t1)^2:
```

```
A20_C:=(x1,y1,t1)->0:
Bc20:=(x1, y1, t1)->(1/2)*k*Bc(x1, y1, t1)^2:
Ac20_C:=(x1,y1,t1)->0:

#O(eps^3)
#lap3
#A3 :=(z,x1,y1,t1)->A3_C(x1,y1,t1)*exp(3*k*z):
#A21:=(z,x1,y1,t1)->A21_C(x1,y1,t1)*exp(2*k*z):
A12:=(z,x1,y1,t1)->A12_C(x1,y1,t1)*exp(k*z)
   +(1/2*I)*z*((diff(B(x1, y1, t1), x1, x1)
   +diff(B(x1, y1, t1), y1, y1))*k^2
   -2*ky^2*(diff(B(x1, y1, t1), y1, y1))
   -4*kx*ky*(diff(B(x1, y1, t1), y1, x1))
   -2*kx^2*(diff(B(x1, y1, t1), x1, x1))
   +(ky^2*(diff(B(x1, y1, t1), y1, y1))
   +2*kx*ky*(diff(B(x1, y1, t1), y1, x1))
   +kx^2*(diff(B(x1, y1, t1), x1, x1)))*k*z)
   *omega*exp(k*z)/k^4:
varphi_bar1:=(z,x1,y1,z1,t1)->0:

#Ac3 :=(z,x1,y1,t1)->Ac3_C(x1,y1,t1)*exp(3*k*z):
#Ac21:=(z,x1,y1,t1)->Ac21_C(x1,y1,t1)*exp(2*k*z):
Ac12:=(z,x1,y1,t1)->Ac12_C(x1,y1,t1)*exp(k*z)
   -(1/2*I)*z*((diff(Bc(x1, y1, t1), x1, x1)
   +diff(Bc(x1, y1, t1), y1, y1))*k^2
   -2*ky^2*(diff(Bc(x1, y1, t1), y1, y1))
   -4*kx*ky*(diff(Bc(x1, y1, t1), y1, x1))
   -2*kx^2*(diff(Bc(x1, y1, t1), x1, x1))
   +(ky^2*(diff(Bc(x1, y1, t1), y1, y1))
   +2*kx*ky*(diff(Bc(x1, y1, t1), y1, x1))
   +kx^2*(diff(Bc(x1, y1, t1), x1, x1)))*k*z)
   *omega*exp(k*z)/k^4:

#dyn3
#A3_C:=(x1,y1,t1)->(1/24*I)*(3*k^2*B(x1, y1, t1)^3
   -8*B3(x1, y1, t1))*omega/k:
#A21_C:=(x1,y1,t1)->-(1/4)*omega*((2*I)*B21(x1,y1,t1)*k
   -ky*B(x1, y1, t1)*(diff(B(x1, y1, t1), y1))
   -kx*B(x1, y1, t1)*(diff(B(x1, y1, t1), x1)))/k^2:
A12_C:=(x1,y1,t1)->
   -(3/8*I)*k*B(x1, y1, t1)^2*Bc(x1, y1, t1)*omega
   +I*(diff(B(x1, y1, t1), t1, t1))/(omega*k):
#vareta_bar1:=(x1, y1, t1)->
   (1/2*I)*omega*Bc(x1, y1, t1)*(diff(B(x1,y1,t1),t1))/g
```

```
246    +(1/4*I)*kx*Bc(x1, y1, t1)*(diff(B(x1, y1, t1),x1))/k
247    -(1/4*I)*kx*B(x1, y1, t1)*(diff(Bc(x1, y1, t1),x1))/k
248    -(1/4*I)*ky*B(x1, y1, t1)*(diff(Bc(x1, y1, t1),y1))/k
249    -(diff(varphi0_C(x1, y1, z1, t1), t1))/g
250    +(1/4*I)*ky*Bc(x1, y1, t1)*(diff(B(x1, y1, t1),y1))/k
251    -(1/2*I)*omega*B(x1,y1,t1)*(diff(Bc(x1,y1,t1),t1))/g:
252
253 #Ac3_C:=(x1,y1,t1)->-(1/24*I)*(3*k^2*Bc(x1, y1, t1)^3
254    -8*Bc3(x1, y1, t1))*omega/k:
255 #Ac21_C:=(x1,y1,t1)->(1/4)*omega*(2*I*Bc21(x1,y1,t1)*k
256    +ky*Bc(x1, y1, t1)*(diff(Bc(x1, y1, t1), y1))
257    +kx*Bc(x1, y1, t1)*(diff(Bc(x1, y1, t1), x1)))/k^2:
258  Ac12_C:=(x1,y1,t1)->
259    (3/8*I)*k*Bc(x1, y1, t1)^2*B(x1, y1, t1)*omega
260    -I*(diff(Bc(x1, y1, t1), t1, t1))/(omega*k):
261
262 #kin3
263 #A3_C:=(x1,y1,t1)->(1/8*I)*omega*(3*k^2*B(x1, y1, t1)^3
264    -8*B3(x1, y1, t1))/k:
265 #A21_C:=(x1,y1,t1)->
266    (k^2*B(x1, y1, t1)*(diff(B(x1, y1, t1), t1))
267    -I*B21(x1, y1, t1)*omega*k
268    +B(x1, y1, t1)*omega*kx*(diff(B(x1, y1, t1), x1))
269    +B(x1, y1, t1)*omega*ky*(diff(B(x1,y1,t1),y1)))/k^2:
270 #A12_C:=(x1,y1,t1)->(-(1/2I)*omega/k^3
271    +I*omega*kx^2/k^5)*(diff(B(x1, y1, t1), x1, x1))
272    +(-(1/2*I)*omega/k^3
273    +I*omega*ky^2/k^5)*(diff(B(x1, y1, t1), y1, y1))
274    +(5/8*I)*omega*k*B(x1, y1, t1)^2*Bc(x1, y1, t1)
275    +(2*I)*omega*kx*ky*(diff(B(x1, y1, t1), y1, x1))/k^5:
276
277 #Ac3_C:=(x1,y1,t1)->-1/8*I*omega*(3*k^2*Bc(x1,y1,t1)^3
278    -8*Bc3(x1, y1, t1))/k:
279 #Ac21_C:=(x1,y1,t1)->
280    (k^2*Bc(x1, y1, t1)*(diff(Bc(x1, y1, t1), t1))
281    +I*Bc21(x1, y1, t1)*omega*k
282    +Bc(x1, y1, t1)*omega*kx*(diff(Bc(x1, y1, t1), x1))
283    +Bc(x1,y1,t1)*omega*ky*(diff(Bc(x1,y1,t1), y1)))/k^2:
284 #Ac12_C:=(x1,y1,t1)->((1/2*I)*omega/k^3
285    -I*omega*kx^2/k^5)*(diff(B(x1, y1, t1), x1, x1))
286    +((1/2*I)*omega/k^3
287    -I*omega*ky^2/k^5)*(diff(B(x1, y1, t1), y1, y1))
288    -(5/8*I)*omega*k*B(x1, y1, t1)^2*Bc(x1, y1, t1)
289    -(2*I)*omega*kx*ky*(diff(B(x1, y1, t1), y1, x1))/k^5:
```

```
290
291 #surf3
292 A3 :=(z,x1,y1,t1)->0:
293 B3 :=(x1,y1,t1)->(3/8)*k^2*B(x1, y1, t1)^3:
294 B21:=(x1,y1,t1)->
295    I*k/omega*B(x1,y1,t1)*diff(B(x1,y1,t1),t1):
296 A21:=(z,x1,y1,t1)->0:
297 vareta_bar1:=(x1, y1, t1)->
298    -(diff(varphi0_C(x1, y1, z1, t1), t1))/g:
299
300 Ac3 :=(z,x1,y1,t1)->0:
301 Bc3 :=(x1,y1,t1)->(3/8)*k^2*Bc(x1, y1, t1)^3:
302 Ac21:=(z,x1,y1,t1)->0:
303
304 #O(eps^4)
305 A13:=(z,x1,y1,t1)->A13_C(x1,y1,t1)*exp(k*z)
306    -((-(1/2)*z^2*exp(k*z)*kx*g/(k^2*omega)
307    -(3/4)*z*exp(k*z)*kx^3*omega/k^6
308    +(3/4)*z^2*exp(k*z)*kx^3*omega/k^5
309    -(1/6)*z^3*exp(k*z)*kx^3*omega/k^4
310    +z*exp(k*z)*kx*g/(k^3*omega))
311    *(diff(B(x1 ,y1 ,t1), x1, x1, x1))
312    +(z*exp(k*z)*ky*g/(k^3*omega)
313    -(1/6)*z^3*exp(k*z)*ky^3*omega/k^4
314    -(1/2)*z^2*exp(k*z)*ky*g/(k^2*omega)
315    -(3/4)*z*exp(k*z)*ky^3*omega/k^6
316    +(3/4)*z^2*exp(k*z)*ky^3*omega/k^5)
317    *(diff(B(x1, y1, t1), y1, y1, y1))
318    -z*exp(k*z)*kx
319    *(diff(B(x1,y1,t1),x1,t1,t1))/(k^2*omega)
320    -z*exp(k*z)*ky
321    *(diff(B(x1,y1,t1),y1,t1,t1))/(k^2*omega)
322    +((9/4)*z^2*exp(k*z)*ky*omega*kx^2/k^5
323    -(9/4)*z*exp(k*z)*ky*omega*kx^2/k^6
324    +z*exp(k*z)*ky*g/(k^3*omega)
325    -(1/2)*z^3*exp(k*z)*ky*omega*kx^2/k^4
326    -(1/2)*z^2*exp(k*z)*ky*g/(k^2*omega))
327    *(diff(B(x1, y1, t1), y1, x1, x1))
328    +(-(1/2)*z^2*exp(k*z)*kx*g/(k^2*omega)
329    -(1/2)*z^3*exp(k*z)*ky^2*omega*kx/k^4
330    +(9/4)*z^2*exp(k*z)*ky^2*omega*kx/k^5
331    +z*exp(k*z)*kx*g/(k^3*omega)
332    -(9/4)*z*exp(k*z)*ky^2*omega*kx/k^6)
333    *(diff(B(x1, y1, t1), y1, y1, x1))
```

```
334   +(3/8)*z*k*exp(k*z)*kx*B(x1, y1, t1)^2
335   *(diff(Bc(x1, y1, t1), x1))*g/omega
336   +(3/4)*z*k*exp(k*z)*kx*B(x1, y1, t1)*Bc(x1, y1, t1)
337   *(diff(B(x1, y1, t1), x1))*g/omega
338   +(1/2)*z*exp(k*z)*(diff(B(x1,y1,t1),x1,x1,t1))/k^2
339   +(3/4)*z*k*exp(k*z)*ky*B(x1, y1, t1)*Bc(x1, y1, t1)
340   *(diff(B(x1, y1, t1), y1))*g/omega
341   +(1/2)*z*exp(k*z)*(diff(B(x1,y1,t1), y1, y1, t1))/k^2
342   +(3/8)*z*k*exp(k*z)*ky*B(x1, y1, t1)^2
343   *(diff(Bc(x1, y1, t1), y1))*g/omega):
```

Bibliography

Mary L. Boas. *Mathematical Methods in the Physical Sciences*. John Wiley & Sons, third edition, 2006. ISBN 0-471-19826-9.

Ronald N. Bracewell. *The Fourier transform and its applications*. McGraw Hill, third edition, 2000. ISBN 0-07-303938-1.

Bjørn Gjevik, Geir K. Pedersen, and Karsten Trulsen. *Forelesninger i Bølgeteori*. Unipub AS, 2007.

Chiang C. Mei. *The Applied Dynamics of Ocean Surface Waves*. World Scientific Pub Co Inc, 1989.

William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes*. Cambridge University Press, New York, NY, USA, third edition, 2007. ISBN 0521884071.

Karsten Trulsen. Weakly nonlinear and stochastic properties of ocean wave fields. application to an extreme wave event. *Waves in geophysical fluids: Tsunamis, rogue waves, internal waves and internal tides*, pages 49–106, 2006.

Karsten Trulsen. On weakly nonlinear modulation of waves on deep water. *Physics of Fluids*, 12(10):2432, 2000.